
175 PTAS
72
mi computer
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
EPSON
CX-21
ON
OFF
POWER
READY
Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen VI-Fascículo 72

Director:	José Mas Godayol
Director editorial:	Gerardo Romero
Jefe de redacción:	Pablo Parra
Coordinación editorial:	Jaime Mardones
	Francisco Martín
Asesor técnico:	Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.º, 08008 Barcelona
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-034-7 (tomo 6)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 298505

Impreso en España-Printed in Spain-Mayo 1985

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.º 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.º 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 19 425 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 429 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S.A. (Paseo de Gracia, 88, 5.º, 08008 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.





Guía del comprador

¿Cuáles son las preguntas que el usuario debería formular para adquirir el equipo más adecuado? Aquí se las sugerimos

Cuando usted eligió su ordenador, quizá fue aconsejado en el sentido de que decidiera primero el software que deseaba utilizar y optase luego por el hardware necesario para ejecutarlo. Hasta cierto punto, éste es un buen consejo cuando se trata de elegir un sistema para comunicaciones. Sin embargo, creemos que a menudo es muchísimo mejor adquirir el modem y el software como paquete global en el mismo comercio.

Lo primero que debe hacer es decidir para qué utilizará el sistema. Para acceder a sistemas de videotexto tales como el Prestel necesitará un modem de 1200/75 baudios; para la mayoría de los sistemas de tablones de anuncios y correo electrónico precisará un modem de 300 baudios, y para el trabajo directo usuario-usuario se recomienda una velocidad de 1200 baudios. Para acceder a Compunet se requiere un modem Compunet especial, puesto que el software está almacenado en la ROM del modem. Usted tendrá, asimismo, que tener en cuenta las frecuencias utilizadas. En Gran Bretaña y España, por ejemplo, necesitará frecuencias CCITT; en Estados Unidos se emplean tonos Bell. Por consiguiente, si desea realizar llamadas transoceánicas directas, es preciso que su modem posea ambas frecuencias.

Para las comunicaciones usuario-usuario es sumamente aconsejable tener un modem que pueda conmutar entre frecuencias de emisión y de recepción. Si éste opera sólo en la de emisión, el modem al cual llame deberá ser capaz de conmutarse a la de respuesta.

Si pretende llamar a tablones de anuncios, resulta esencial un modem con la facilidad de *llamada automática* (no disponible en España). Ello se debe a que la mayoría de los tablones de anuncios poseen únicamente una línea telefónica y sólo permiten el acceso a un usuario a la vez. Por este motivo con frecuencia están comunicando, de modo que es lógico dejar que el modem realice la llamada repetidamente por usted. Un modem con llamada automática también debe soportar software capaz de obtener un número (ya sea desde el teclado o bien desde una base de datos de números de teléfono) y enviarlo al modem en el formato correcto. Lamentablemente, los diferentes modems con llamada automática esperan que el número que se les envíe esté en formatos diferentes, de modo que es necesario que el modem y el software sean compatibles: un buen argumento para adquirir tanto el modem como el software en un solo paquete.



Steve Cross

Unidad ideal
La gama disponible de opciones de hardware y software hacen que adquirir un modem se convierta en una tarea confusa y especializada: nuestra unidad ideal (reflejada en la ilustración) combina las características de los numerosos modems que existen en la actualidad. Para un principiante, el mejor enfoque es especificar por anticipado sus exigencias en cuanto a comunicaciones y confiar en el vendedor para que elija por él el paquete adecuado

Si desea que la gente le envíe datos directamente, hallará muy justificada la inversión que supone la adquisición de un modem con *contestador automático*. No obstante, si piensa utilizar para este fin su línea telefónica normal, deberá tener la cortesía de advertírselo a sus amigos, en especial a aquellos que no posean un modem. De lo contrario, ¡su modem podría silbarles durante diez segundos y después colgar!

El software adecuado también es esencial para un modem con contestador automático. Esta clase de software comprende desde paquetes capaces de abrir un nuevo archivo para cada llamada y guardarlo en disco, hasta software sofisticado para tablones de anuncios tales como el TBBS. Un interesante ejemplo de software con contestador automático para micros CP/M es el Remote CP/M. Este paquete permite que usted "disque" su micro CP/M y ejecute cualquier programa CP/M a través del te-



Protek 1200
Tipo: Modem acústico
Velocidad: 1200/75 y 1200/1200 baudios



Prism 1000
Tipo: Modem compacto para videotexto
Velocidad: 1200/75 baudios



Commodore Communications Modem
Tipo: Modem compacto de videotexto para Compunet
Velocidad: 1200/75 y 1200/1200 baudios



Ian McKinnell

Epson CX-21
Tipo: Modem acústico
Velocidad: 300 baudios

léfono, lo que resulta ideal para aquellos usuarios que posean un micro de sobremesa o uno portátil.

Al elegir el software apropiado, es casi seguro que desee un paquete que sea capaz de recibir y emitir archivos ASCII. Asegúrese de que el paquete soporte cualquier dispositivo de almacenamiento que usted utilice. Los programas en BASIC se pueden transmitir en forma ASCII, como ya sabemos; pero si desea transmitir archivos binarios (p. ej., archivos .COM CP/M), necesitará alguna clase de protocolo para transmisión binaria. De éstos, el más difundido es el XModem.

También es conveniente poder crear archivos de *conexión automática* para distintos sistemas. Entonces, cuando conecte con un sistema, todo cuando deberá hacer es cargar el archivo apropiado, conteniendo su identificador, contraseña, etc. Algunos sistemas con llamada automática enlazarán este tipo de archivos con una base de datos de números de teléfono, de modo que lo único que ha de hacer es entrar el nombre del servicio que desea; el software buscará el número de teléfono, lo marcará y se conectará automáticamente.

Una vez decidido respecto a las características que requiere, ha de encontrar un paquete de modem y software que soporte estas facilidades. Puede adquirir el modem y el software por separado, pero le aconsejamos decididamente que le proporcione al vendedor una lista de las características que desea, así como detalles sobre el micro que utilizará, y deje que él le busque un paquete completo compuesto por modem, cable y software. De esta forma, si el sistema no cumple con lo que deseaba que hiciera, tanto usted como el comerciante sabrán quien debe arreglar la situación.

Elección de un terminal idóneo

Si ya posee un micro, es probable que desee usarlo como terminal. Ello será factible independientemente de la máquina que posea (si está muy decidido, incluso se puede utilizar un ZX81), si bien algunos micros se prestan mejor que otros para las comunicaciones. He aquí un breve resumen de la idoneidad de los cuatro micros más populares.

Con mucho, la máquina más fácil de convertir en terminal es el BBC Micro. De hecho, usted puede escribir un simple programa de terminal mudo para el mismo con unas pocas líneas de BASIC:

```
100 REM Programa de terminal mudo para el BBC
110 *FX2,2
120 *FX3,1
130 REPEAT: GET A$: IFA$=CHR$(13) THEN PRINT
140 PRINT A$;:UNTIL FALSE
```

Para el BBC Micro existen varios buenos paquetes de comunicaciones, algunos de los cuales se suministran en ROM, pero suelen ser caros. La mayoría ofrecen todas las características requeridas por el usuario medio, porque la mayor parte del trabajo lo realiza el sistema operativo del ordenador: todo lo que ha de hacer el programador es añadir los toques finales.

El Spectrum es más difícil de adaptar. En primer lugar, no podrá batir ningún récord de velocidad en materia de comunicaciones desde BASIC. Con un programa en este lenguaje apenas es posible empujar el Spectrum hasta unos 10 baudios y entonces no será capaz de llevar a cabo tareas tales como almacenar los caracteres en RAM. También puede olvidarse de su Interface 1: no es una interface RS232 y sirve de poco para comunicaciones. Para el Spectrum todavía no existe virtualmente software para comunicaciones.

El Commodore 64 también posee una interface en serie no estándar y la mayoría de los modems para la máquina se enchufan en la puerta para el usuario. Nuevamente, tampoco podrá hacer nada muy útil en BASIC. Asimismo, el 64 posee un juego de caracteres ASCII no estándar, de modo que el software para comunicaciones ha de traducir entre ASCII estándar y ASCII Commodore, algo que se puede hacer muy fácilmente empleando una tabla de referencia. No existen paquetes de software de comunicaciones aprobados por Commodore. Los usuarios británicos tienen la posibilidad de obtener el Termulator de Chris Townsend Computers; los norteamericanos habrán de consultarlo con las tiendas locales. Los usuarios de Compunet pueden utilizar el modem Compunet Commodore oficial con software residente, pero este modem es exclusivo para Compunet.

Tandy suministra software de terminal mudo para sus máquinas TRS-80 basadas en disco que operan bajo la mayoría de los sistemas operativos. La mayor parte de este software está destinado a la transferencia directa máquina-máquina, pero también se puede emplear con modems. Las máquinas Tandy fueron los primeros micros que se utilizaron para ejecutar y acceder a tablones de anuncios, de modo que normalmente se puede hallar una buena selección de software para comunicaciones de dominio público, basado tanto en disco como en cassette. Las TRS-80 no son aptas para operación de videotexto (1200/75 baudios).

En realidad cualquier micro de gestión se puede utilizar para comunicaciones, pero existen varios puntos importantes a tener en cuenta. En primer lugar, existe una creciente tendencia hacia los modems incorporados; éstos (en particular aquellos con software de comunicaciones basado en ROM) son los más fáciles de emplear. Tan sólo es necesario enchufarlos en el conector del teléfono.

Si se desestima la opción del modem incorporado, la siguiente mejor opción es un micro con al menos dos puertas RS232, que permitirá el uso simultáneo de un modem y una impresora en serie. Algunos micros poseen puertas para modem separadas y no estándares: éstas servirán siempre y cuando usted pueda conseguir un cable adecuado para el modem que elija.

Desde el punto de vista del software, no tendrá ningún problema con máquinas CP/M, MS-DOS o PC-DOS. Los sistemas operativos no estandarizados, sin embargo, corren tanto riesgo desde el punto de vista de los paquetes para comunicaciones como con cualquier otro tipo de software.

Si las comunicaciones son su principal razón para adquirir un micro, las llamadas máquinas "de regazo", tales como el Tandy Modelo 100, el NEC PC8201A y el Olivetti M10 son muy interesantes. Con una de ellas y un modem que funcione con pilas tendrá un terminal de maletín convenientemente portátil. Las tres máquinas poseen editores de texto incorporados y software de terminal y permiten unas 20 horas de uso con cuatro pilas AA.



Tomar medidas

Diseñaremos un fragmento de software para que el robot pueda ubicar y medir con exactitud el lado de un objeto

Conseguir que nuestro robot localice y mida el lado de un objeto exige un software algo complejo. El robot investigará el objeto empleando los sensores de microinterruptor que ya le hemos instalado. Las primeras ideas que tuvimos para un posible método de realizar esta tarea fueron:

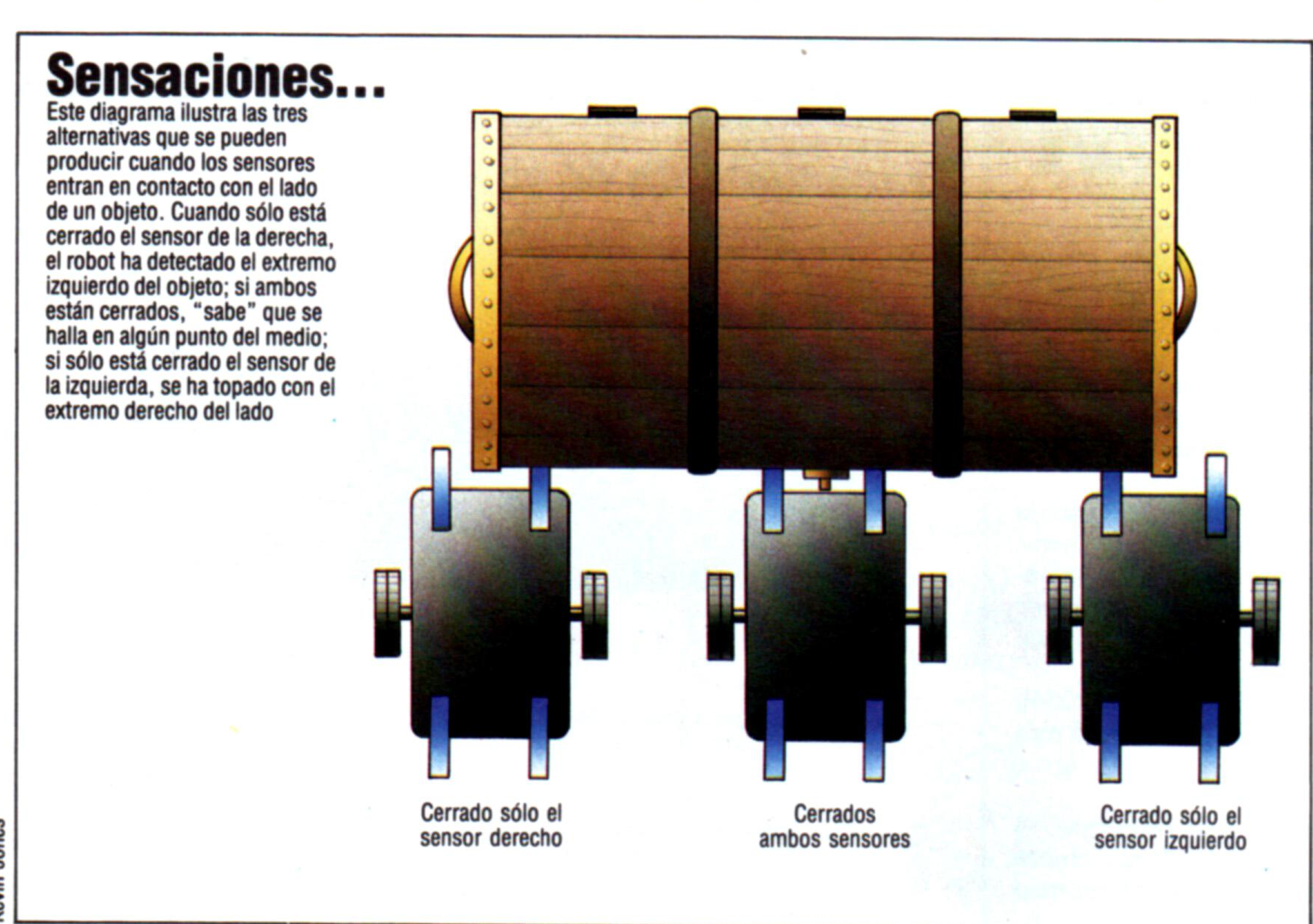
1) Hallar el objeto.
2) Encontrar un extremo del lado localizado.
3) Ir probando a lo largo del lado del objeto hasta hallar el otro extremo.

La primera etapa se puede cubrir fácilmente si suponemos que cuando comienza el programa el robot está orientado hacia el lado del objeto que deseamos medir. El principal problema que se puede prever es que el robot alcance uno de los extremos del lado con un solo sensor en vez de hacer contacto con ambos. En el diagrama se reflejan las posibles variaciones. No obstante, aun cuando uno de los sensores delanteros estuviera cerrado, se puede saber si se trata del izquiedo o del derecho y, por consiguiente, podemos desarrollar una estrategia para tratar esta situación.

Debemos, asimismo, dar por sentado que inicialmente el robot está posicionado en 90° respecto al lado del objeto a medir. Con ello nos ahorramos el tratar casos en los cuales el robot efectúe un contacto oblicuo con el objeto.

La segunda etapa de nuestro método se simplifica decidiendo que el robot siempre avance hacia el extremo derecho del objeto antes de comenzar a medirlo. Para localizar el extremo derecho, el robot debe "sentir" su recorrido a lo largo del lado, desplazándose en pasos discretos hacia la derecha hasta que sólo se cierra el sensor izquierdo (y no ambos). Para ir sondeando el lado, el robot ha de llevar a cabo una serie complicada de maniobras, implicando cada paso cinco movimientos. Suponiendo que inicialmente el robot esté en contacto con el lado del objeto, debe primero retroceder, luego girar 90°, avanzar una cierta distancia, volver a girar 90° y por último avanzar hasta que los sensores vuelvan a hacer contacto con el objeto. El diagrama ilustra los pasos que supone la maniobra completa. La *longitud de paso* (distancia entre un punto de contacto con el lado del objeto y el siguiente punto) es equivalente a la parte de la maniobra en la cual el robot se desplaza paralelamente al lado del objeto.

Para localizar con precisión el extremo derecho del objeto, podría parecer necesario que el robot sondeara el lado en pasos de unos pocos milímetros, pero no es así. Por el contrario, podemos utilizar pasos mayores, sondeando el lado del objeto hasta sobrepasar el extremo, y luego retroceder un paso para sondear en pasos menores para localizar



Sensaciones...
Este diagrama ilustra las tres alternativas que se pueden producir cuando los sensores entran en contacto con el lado de un objeto. Cuando sólo está cerrado el sensor de la derecha, el robot ha detectado el extremo izquierdo del objeto; si ambos están cerrados, "sabe" que se halla en algún punto del medio; si sólo está cerrado el sensor de la izquierda, se ha topado con el extremo derecho del lado

Kevin Jones

con exactitud el extremo derecho. Una longitud de paso adecuada es la distancia entre los dos sensores delanteros (alrededor de 60 mm), puesto que ello asegura que cuando se sobrepase un extremo, uno de los sensores se cierre.

La tercera etapa supone un procedimiento de sondeo similar, pero en esta ocasión el robot se desplaza hacia la izquierda, contando la cantidad de pasos dados hasta llegar al extremo izquierdo. Al final de esta etapa, la longitud del lado del objeto estará almacenada en la variable del contador y podrá ser impresa.

Programa de medición

Ofrecemos listados para el Commodore 64 y el BBC Micro. Se deben insertar los coeficientes impulso/distancia e impulso/ángulo para su propio robot (hallados mediante la experimentación en el capítulo anterior del proyecto del robot). La estructura de procedimientos del BASIC BBC es ideal para escribir un programa de este tipo. Podemos adoptar un enfoque sumamente estructurado para este problema, controlando, mediante procedimientos separados, cada movimiento que efectúe el robot. Dos características del BASIC BBC (los nombres de variables ampliados y el paso de parámetros entre procedimientos) hacen que el programa se asemeje mucho a la forma en que nosotros pensamos. La versión Commodore puede adoptar el mismo enfoque estructurado, pero su estructuración en BASIC resulta mucho más dificultosa, lo que hace que el programa sea mucho más difícil de seguir que la versión para el BBC.

Habiendo aislado las principales tareas que debe llevar a cabo el programa, podemos diseñar procedimientos individuales para desplazar el robot y combinar una serie de maniobras para crear un método de *sondeo*. La combinación de los procedimientos de sondeo forma el procedimiento de *medición* global. En esta aplicación, los distintos niveles de procedimientos se pueden identificar fácilmente, yendo desde un sencillo método para impulsar los motores, en el nivel inferior, hasta la actividad de medición completa en el nivel más elevado.

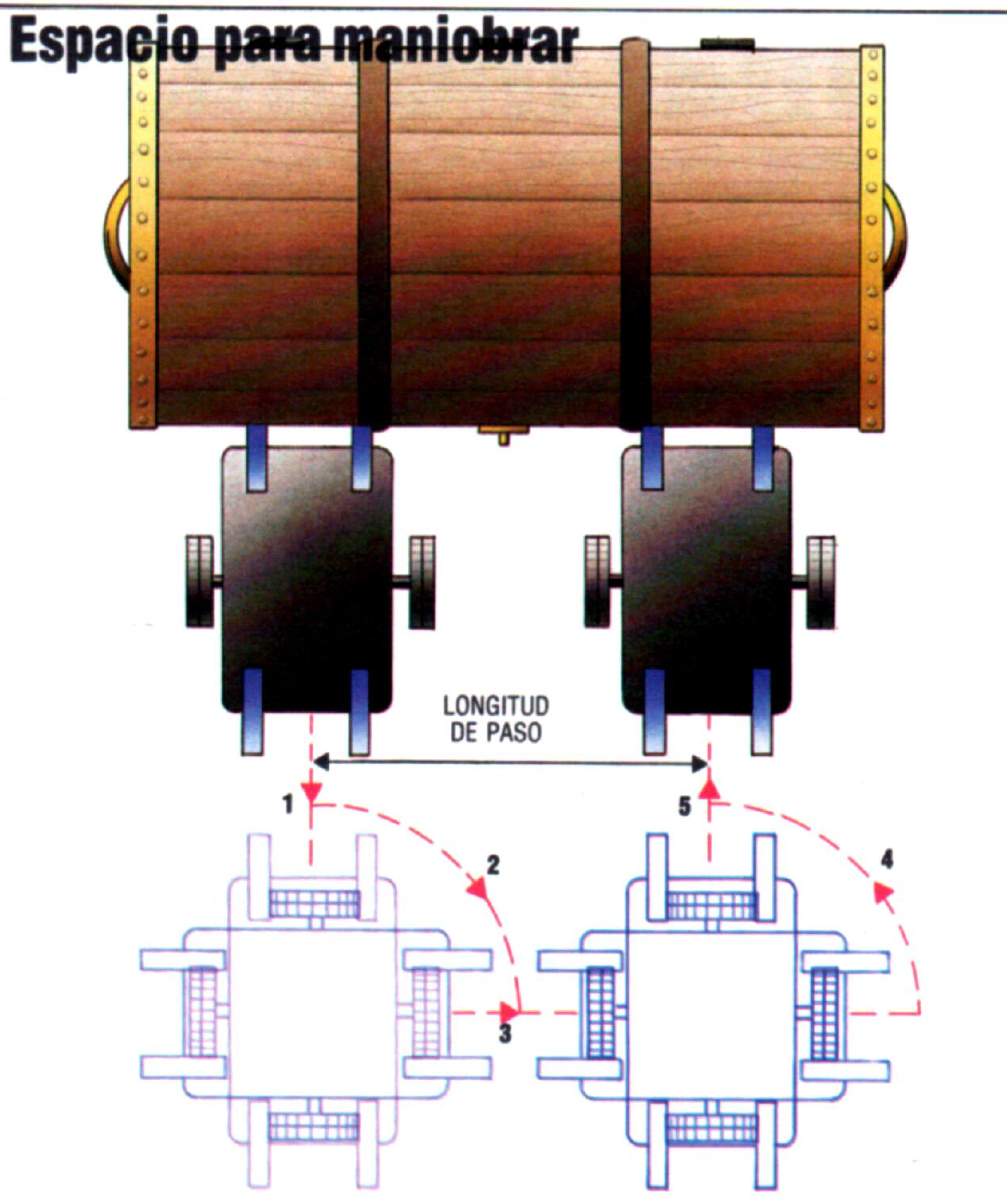
Si inicialmente no se coloca el robot exactamente en 90° respecto al lado del objeto a medir, pueden surgir problemas. Si sólo uno de los sensores hace contacto cuando ambos deberían hacerlo, entonces la lógica del programa hará que el robot decida que está posicionado en uno de los extremos del objeto. Si esto sucede, interrumpa el programa, alinee al robot perpendicularmente al lado a medir, y vuelva a ejecutar el programa desde el principio.

Se pueden identificar varios errores intrínsecos de medición. La anchura de cada sensor, por ejemplo, es de unos 5 mm. La localización de los extremos izquierdo y derecho del objeto puede, por consiguiente, producir un error máximo total de 10 mm.

Además, cuando el robot sondea con precisión los extremos del objeto, lo hace en pasos de 5 mm y, por lo tanto, se puede introducir aún otro error de 10 mm.

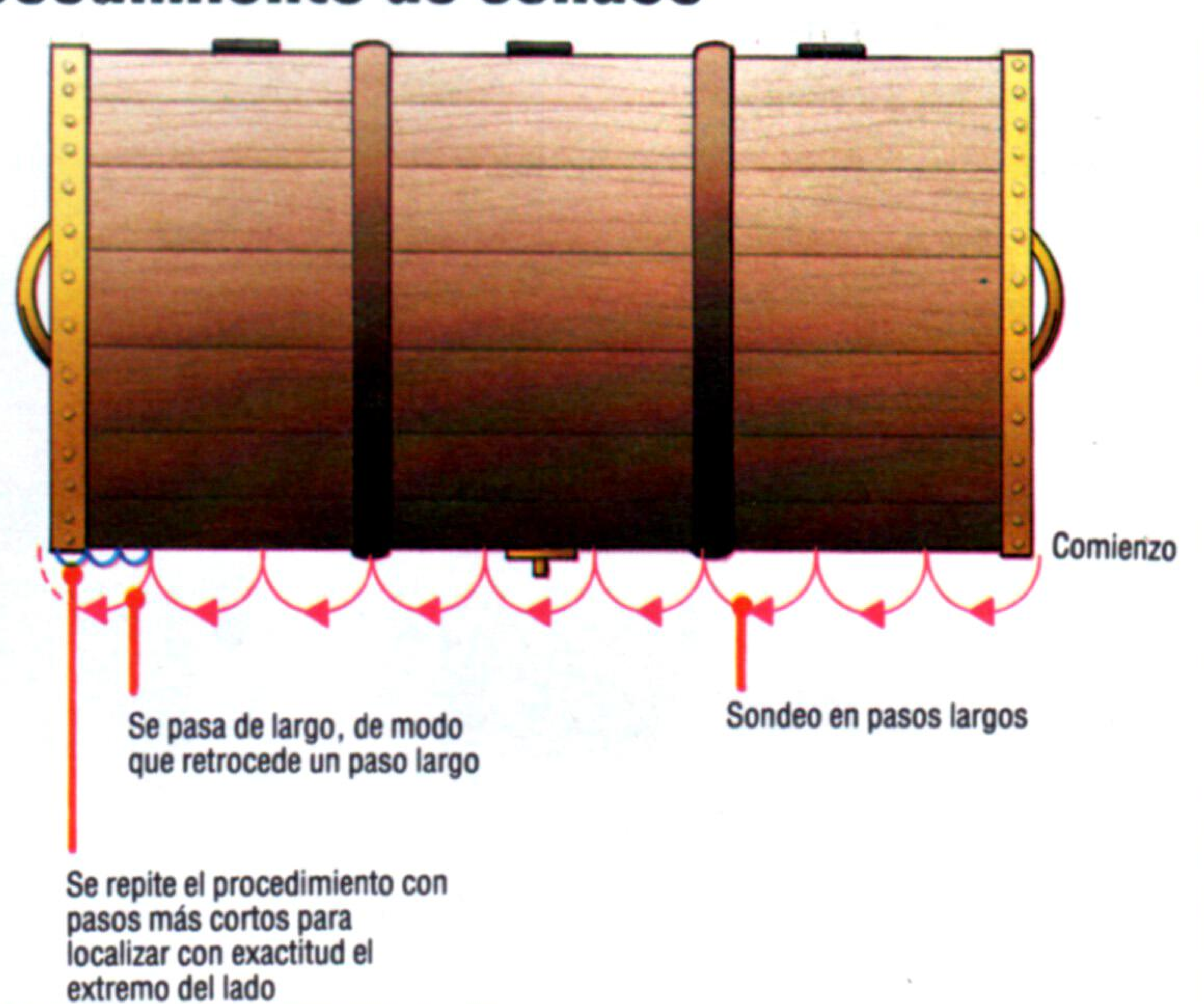
En las pruebas con nuestro robot prototipo, el promedio de error para la medición de un objeto de 410 mm de lado fue de unos 20 mm: un margen de sólo el 5 %.

Espacio para maniobrar

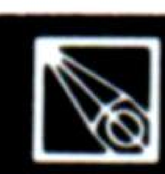


La figura superior muestra la maniobra de sondeo básica del robot. Cuando ambos sensores están cerrados, el robot retrocede (1) para poder girar sin chocar contra el objeto.
El robot efectúa luego un giro de 90° (2) y se desplaza hacia adelante cubriendo la longitud de paso (3). El robot realiza otro giro de 90°, de modo que queda nuevamente encarado al objeto (4) y, por último, avanza (5) para comprobar si el objeto se halla todavía enfrente de él.
Este proceso se repite en toda la longitud del objeto, como podemos ver abajo. Al principio los pasos del robot son largos, para obtener una longitud aproximada del objeto. Cuando sólo uno de los sensores, o ninguno, está cerrado, el robot desanda el último paso y repite en su totalidad el procedimiento de sondeo con pasos más cortos

Procedimiento de sondeo



Kevin Jones



Listado para BBC Micro

```
1000 REM **** MEDICION ROBOT BBC ****
1010 MODE 7
1020 PROCinicializar
1030 PROCmedir
1040 PROCimprimir
1050 END
1060 DEF PROCmedir
1070 PROChallar
1080 REM ** SOLO UN PARACHOQUES ? **
1090 PROCcomprobar_parachoques
1100 REM **** HALLAR EXTREMO ****
1110 REPEAT:PROCsondear(derecha,anchura)
1120 UNTIL (?REGDAT AND 192)=parachoques_izquierdo
1130 REM ** RETROCEDER Y AVANZAR HASTA EL EXTREMO **
1140 PROCsondear(izquierda, anchura)
1150 REPEAT:PROCsondear(derecha,anchura_pequeña)
1160 UNTIL (?REGDAT AND 192)=parachoques_izquierdo
1170 PRINT"HALLADO EXTREMO DERECHO"
1180 PRINT"COMIENZA LA MEDICION"
1190 REM ** EMPEZAR A MEDIR **
1200 contador=anchura
1210 REPEAT:PROCsondear(izquierda,anchura)
1220 contador=contador+anchura
1230 UNTIL (?REGDAT AND 192)=parachoques_derecho
1240 REM ** RETROCEDER Y AVANZAR HASTA EL EXTREMO **
1250 contador=contador-anchura
1260 PROCsondear(derecha,anchura)
1270 REPEAT:PROCsondear(izquierda,anchura_pequeña)
1280 contador=contador+anchura_pequeña
1290 UNTIL (?REGDAT AND 192)=parachoques_derecho
1300 ?REGDAT=0
1310 ENDPROC
1320 :
1330 DEF PROCimprimir
1340 CLS
1350 PRINTTAB(5,12)"EL LADO DEL OBJETO
     MIDE   ";contador;"  mm"
1360 ENDPROC
1370 :
1380 DEF PROCinicializar
1390 RDD=&FE62:REGDAT=&FE60
1400 ?RDD=15:REM LINEAS 0-3 SALIDA
1410 ?REGDAT=1:REM ENCENDER BIT PUESTA A CERO
1420 adelante=4:atras=2:izquierda=6:derecha=0
1430 coeficiente_id=3.34446:coeficiente_ia=375/90
1440 parachoques_derecho=128:parachoques_izquierdo=64
1450 ambos_parachoques=0:ningun_parachoques=192
1460 anchura=60:anchura_pequeña=5
1470 ENDPROC
1480 :
1490 DEF PROCbuscar(sentido)
1500 REPEAT:PROCsondear(sentido,anchura)
1510 UNTIL (?REGDAT AND 192)=ambos_parachoques
1520 ENDPROC
1530 :
1540 DEF PROChallar
1550 REPEAT:PROCmover(adelante,8)
1560 UNTIL (?REGDAT AND 192)<>ningun_parachoques
1570 ENDPROC
1580 :
1590 DEF PROCcomprobar_parachoques
1600 IF (?REGDAT AND 192)=parachoques_derecho THEN
     PROCbuscar(derecha):ENDPROC
1610 IF (?REGDAT AND 192)=parachoques_izquierdo THEN
     PROCbuscar(izquierda):ENDPROC
1620 ENDPROC
1630 :
1640 DEF PROCsondear(direccion,paso)
1650 IF direccion=derecha THEN direccion_op=izquierda ELSE
     direccion_op=derecha
1660 PROCmover(atras,30)
1670 PROCgirar(direccion,90)
1680 PROCmover(adelante,paso)
1690 PROCgirar(direccion_op,90)
1700 REPEAT:PROCmover(adelante,8)
1710 UNTIL (?REGDAT AND 192)<>ningun_parachoques
1720 ENDPROC
1730 :
1740 DEF PROCmover(dir,distancia)
1750 ?REGDAT=(?REGDAT AND 1)OR dir
1760 impulsos=coeficiente_id*distancia
1770 FOR I=1 TO impulsos:PROCimpulso:NEXT I
1780 ENDPROC
1790 :
1800 DEF PROCgirar(dir,angulo)
1810 ?REGDAT=(?REGDAT AND 1)OR dir
1820 impulsos=coeficiente_ia*angulo
1830 FOR I=1 TO impulsos:PROCimpulso:NEXT I
1840 ENDPROC
1850 DEF PROCimpulso
1860 ?REGDAT(?REGDAT OR 8)
1870 ?REGDAT=(?REGDAT AND 247)
1880 ENDPROC
```

Listado para Commodore 64

```
10 REM **** MEDICION ROBOT CBM ****
20 GOSUB1000:REM INICIALIZAR
30 GOSUB2000:REM MEDIR
40 GOSUB3000:REM IMPRIMIR
50 END
60 :
1000 REM **** INICIALIZAR ****
1010 RDD=56579: REGDAT=56577
1020 POKE RDD,15:REM LINEAS 0-3 SALIDA
1030 POKE REGDAT,1:REM ENCENDER BIT PUESTA A CERO
1040 FW=4: BW=2: LF=6: RT=0
1050 PD=3.34446:PA=375/90
1060 RB=128:LB=64:BB040:NB=192
1070 WD=80:SW=5
1080 RETURN
1090 :
2000 REM **** MEDIR ****
2010 GOSUB3500:REM HALLAR OBJETO
2020 GOSUB4000:COMPROBAR PARACHOQUES
2030 REM ** HALLAR EXTREMO **
2040 WY=RT:SP=WD:GOSUB6000:REM SONDEAR
2050 IF(PEEK(REGDAT)AND192)<>LB THEN 2040
2060 REM ** RETROCEDER Y AVANZAR HACIA EXTREMO **
2070 DR=LF:DS=WD:GOSUB6000:REM SONDEAR
2080 DR=RT:DS=SW:GOSUB6000:REM SONDEAR
2090 IF(PEEK(REGDAT)AND 192)<>LB THEN 2080
2100 PRINT"HALLADO EXTREMO DERECHO"
2110 PRINT"COMIENZA LA MEDICION"
2120 REM ** EMPEZAR A MEDIR **
2130 CC=WD
2140 DR=LF:DS=WD:GOSUB6000:CC=CC+WD
2150 IF(PEEK(REGDAT)AND 192)<>RB THEN 2140
2160 REM ** RETROCEDER Y AVANZAR HACIA EXTREMO **
2170 CC=CC-WD
2180 DR=RT:DS=WD:GOSUB6000:CC=CC+SW
2190 IF(PEEK(REGDAT)AND 192)<>RB THEN 2180
2200 POKE REGDAT,0
2210 RETURN
2220 :
3000 REM **** IMPRESION ****
3010 PRINTCHR$(147)
3020 PRINT"EL LADO DEL OBJETO MIDE  ";CC; "MM"
3030 RETURN
3040 :
3500 REM **** HALLAR ****
3510 DR=FW:DS=5:GOSUB7000:REM MOVER
3520 IF(PEEK(REGDAT)AND192)=NB THEN 3510
3530 RETURN
3540 :
4000 REM **** COMPROBAR PARACHOQUES ****
4010 IF(PEEK(REGDAT)AND192)=RB THEN
     SS=RT:GOSUB5000:RETURN
4020 IF(PEEK(REGDAT)AND192)=LB THEN
     SS=LF:GOSUB5000:RETURN
4030 RETURN
4040 :
5000 REM **** BUSCAR (SS) ****
5010 DR=FW:DS=8:GOSUB7000:REM MOVER
5020 IF(PEEK(REGDAT)AND192)=NB THEN 5010
5030 RETURN
5040 :
6000 REM **** SONDEAR (WY,SP) ****
6010 IF WY=RT THEN OW=LF
6020 IF WY=LF THEN OW=RT
6030 DR=BW:DS=30:GOSUB7000:REM MOVER
6040 DR=WY:AG=90:GOSUB7500:REM GIRAR
6050 DR=FW:DS=SP:GOSUB7000:REM MOVER
6060 DR=OW:AG=90:GOSUB7500:REM GIRAR
6070 DR=FW:DS=8:GOSUB7000:REM MOVER
6080 IF(PEEK(REGDAT)AND192)=NB THEN 6070
6090 RETURN
6100 :
7000 REM **** MOVER (DR,DS) ****
7010 POKE REGDAT,(PEEK(REGDAT)AND 1)OR DR
7020 PL=PD*DS
7030 FOR I=1 TO PL:GOSUB8000:NEXT I
7040 RETURN
7050 :
7500 REM **** GIRAR (DR,AG) ****
7510 POKE REGDAT,(PEEK(REGDAT)AND 1)OR DR
7520 PL=PA*AG
7530 FOR I=1 TO PL:GOSUB8000:NEXT I
7540 RETURN
7550 :
8000 REM **** IMPULSO ****
8010 POKE REGDAT,PEEK(REGDAT)OR 8
8020 POKE REGDAT,PEEK(REGDAT)AND 247
8030 RETURN
```

Para conocerte mejor

En nuestra serie dedicada al software vertical analizaremos un juego de programas para la "clasificación de la personalidad"

Human Edge Software, una empresa de programación con base en California, ha dado un paso intermedio hacia la inteligencia artificial. Los programas de Human Edge constituyen refinadas herramientas para tomar decisiones comerciales, que trabajan rápidamente a partir de grandes cantidades de información que le proporciona el usuario, evaluando los datos de acuerdo a criterios almacenados y produciendo luego un curso de acción aconsejable. Human Edge es un juego de cuatro programas (*Communication edge, Sales edge, Management edge* y *Negotiation edge*) para máquinas IBM y compatibles. Se afirma que con él se "incrementan las aptitudes profesionales individuales del usuario" en las áreas especificadas en los nombres de cada uno de los programas (comunicaciones, rentas, dirección y negociaciones). Para el Commodore 64, el Apple II y el Macintosh se ha creado una versión reducida, llamada *Mind prober*, que utiliza las mismas técnicas; pero en este capítulo nos centraremos en el juego de cuatro programas.

Se dice que los programas son el producto de más de diez años de desarrollo, suponiendo el trabajo de científicos del comportamiento y expertos en gestión empresarial, e incorporando nuevas técnicas como el análisis de factores humanos, tecnología de sistemas expertos y matemática de teoría de decisiones.

Esta descripción suena bastante técnica y, sumada al costo de los programas, podría intimidar al usuario potencial. No obstante, los programas son fáciles de utilizar y se los puede hacer operar al completo en menos de una hora de autoaprendizaje. Todos ellos se ejecutan a través de menús y están construidos en base a extensos cuestionarios compuestos por una serie de sentencias cuidadosamente redactadas sobre las que se invita al usuario a expresar su conformidad o disconformidad. Las sentencias se dirigen a la evaluación de las características significativas de la personalidad del usuario, así como sus perspectivas de ventas, clientes actuales, subordinados y superiores de la empresa, y cualquier aspecto de las relaciones empresariales que desee investigar el usuario.

El programa evalúa las respuestas y prepara un informe detallado, incluyendo un curso de acción recomendado. Las recomendación puede ser la sugerencia de aproximarse abiertamente a un nuevo cliente, una eficaz estrategia de cierre para una venta difícil o técnicas de negociación a utilizar con empleados o jefes.

Cada programa comienza con un cuestionario de autovaloración que incluye sentencias tales como: "Yo asumo el mando en la mayoría de las reuniones", "Discuto con los demás más que la mayoría", "Soy más bien impulsivo", etc. El usuario decide si la sentencia es una descripción exacta de sí mismo y luego entra su respuesta. La herramienta de autovaloración se ha preparado con gran eficacia, con una considerable superposición de preguntas como medición interna de validez. Por lo tanto, las respuestas del usuario a "Soy más bien impulsivo" y a "A veces actúo sin pensar" se evaluarán una en relación a la otra por razones de coherencia. Completada esta sección, las respuestas se almacenan en disco. Éstas se pueden actualizar y volver a utilizar en cualquier momento.

Después de efectuada la autovaloración, se solicita al usuario que exprese su acuerdo o desacuerdo con una serie de adjetivos relacionados con el objeto de la investigación. Para ayudar al usuario a evaluar a su cliente, su empleado o su jefe, se utilizan palabras tales como hablador, aprensivo, independiente, tenaz, ambicioso, cortés, ostentoso, etc. Al trabajar con esta lista es de gran ayuda tener a mano un buen diccionario de inglés norteamericano para comprender algunos términos, porque el Oxford English Dictionary no reconoce determinados calificativos.

El usuario puede desplazarse por la lista de adjetivos y cambiar sus respuestas en cualquier momento. Al igual que la autovaloración, la lista se guarda en disco, si bien también puede actualizarse según lo dicte la experiencia.

Sólo es necesario completar la autovaloración una vez; luego puede ser recuperada desde el disco y utilizada en relación a cualquiera de las otras lis-


David Higham

tas del "oponente". Se pueden guardar en disco hasta ocho evaluaciones de oponentes. Cuando hay presentes dos listas completas, el programa toma las respuestas y las evalúa, preparando después un informe en el cual se sintetizan las características del usuario y su tema.

El siguiente informe, generado por *Communication edge*, está basado en un usuario real y su oponente (el hijo adolescente del usuario), describiéndose este último como "señor T" (de *test*: prueba). El informe se presenta como si el ordenador le estuviera hablando directamente al usuario:

Para comunicarte con el señor T necesitarás usar tu enfoque flexible y estable de aproximación a la gente. El señor T es una persona muy reservada que prefiere estar sola y tiene muy poca paciencia para mantener charlas y relaciones sociales. Cuando le pidas sus ideas y sus impresiones, espera de él una actitud cínica o sospechosa. Cuando le hables, no supongas por defecto que te está entendiendo. Sé claro, conciso y directo.

Contrastando con tu estilo sereno, el señor T se enfada enseguida e incluso puede parecer enojado antes de que comience la conversación. Podría tratar de imponerte sus propias opiniones. Sé cordial, aunque se porte así. No te extrañes de lo imprevisible de su comportamiento. El señor T puede hablar de forma impulsiva durante un minuto, para, al minuto siguiente, escoger con sumo cuidado cada una de sus palabras. Asume el papel de director del proceso de la conversación. Parafrasea sus comentarios para conseguir claridad y llegar a un acuerdo.

Los padres probablemente reconocerán la descripción de un típico adolescente, si bien *Communication edge* no hace preguntas acerca de la edad del sujeto (sin embargo, sí se tiene en cuenta el sexo).

El vocabulario empleado en los informes les será familiar a quienes lean las columnas de consejos de los periódicos o a quienes hayan participado en tests de personalidad similares. El usuario por lo general se describe en términos amables ("de temperamento sereno, flexible, estable"), mientras que el oponente se ve menos favorecido ("se enfada enseguida, es impredecible, cínico, desconfiado"). Es probable que esto esté diseñado para reforzar la imagen que el usuario tiene de sí mismo y quizá para ajustarse a la forma de pensar de los norteamericanos, en el sentido de considerar los negocios como una guerra, siendo la previsión de ventas o el cliente el enemigo cuya resistencia se debe vencer.

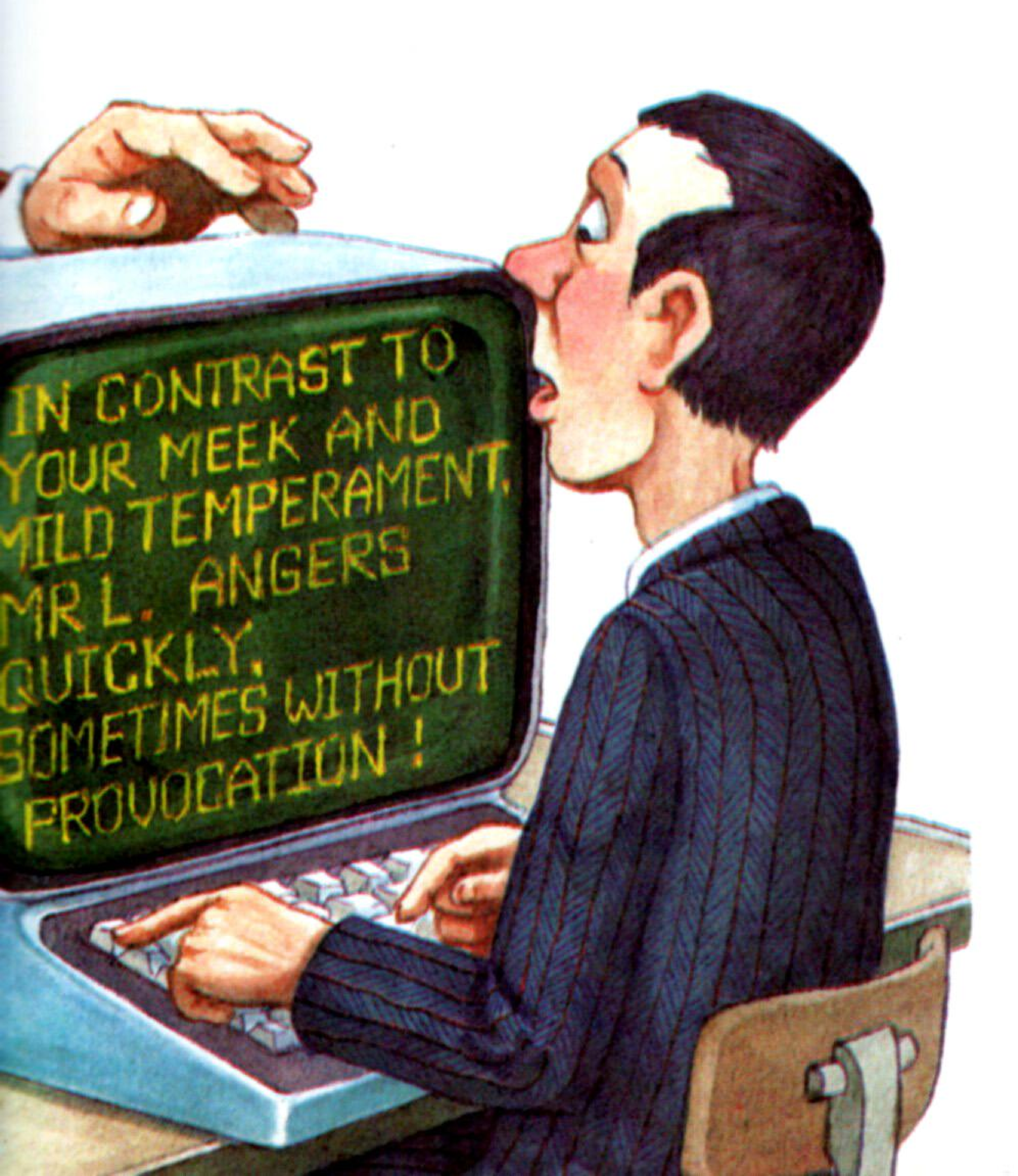
En *Negotiation edge* este enfoque se hace aún más evidente; en el mismo dichas estrategias se recomiendan (impresas en mayúscula) como:

SAQUE PARTIDO DEL CONOCIMIENTO QUE POSEE SOBRE EL SR T
PREPARE AL SR T CON UNAS TEMPRANAS CONCESIONES
ACOSTUMBRE AL SR T A DECIR "SÍ"
ENCUBRA SUS AMENAZAS
EXAGERE SUS PROBLEMAS
SIMULE LA CUANTÍA DE SUS BENEFICIOS
PASE UN BUEN INFORME

Parece ser que quienes desarrollaron los programas asumieron que un usuario sólo tendría uno de los cuatro programas, dado que cada uno de ellos exige que el usuario complete una autovaloración independiente, presentando preguntas similares por un orden ligeramente distinto. A la vista del precio, esta suposición parece razonable; pero puesto que una organización grande probablemente deseará utilizarlos como herramientas para la dirección, hubiera sido más provechoso poder emplear en los cuatro paquetes la misma autovaloración. Pero, dado que los programas son tan parecidos, uno sencillamente puede tomar el menos caro del juego, el *Communication edge*, y adaptar sus informes de modo que se adapten a las diversas situaciones.

Una importante objeción a señalar en los programas, cuando se los va a utilizar como herramientas de dirección, es su incapacidad para aprender a partir de la experiencia, a excepción de la propia actualización que efectúe el usuario de sus valoraciones. Por ejemplo, para éste sería de gran valor poder entrar los resultados de una estrategia propuesta de modo que pudiera modificarla a la luz de la experiencia, especialmente para la evaluación de sujetos de quienes inicialmente se sabe muy poco. Además, muchas de las preguntas son difíciles de responder sobre una base estricta de "estoy de acuerdo o no lo estoy", y no existe evidencia de una estructura arborescente para la formulación de las preguntas, lo que permitiría la ampliación o verificación de las respuestas. Una solución podría ser incluir una opción "no lo sé", que abriría entonces el camino para una pregunta de nivel inferior que se pudiera responder afirmativa o negativamente.

En el análisis final, uno puede creer o no en este tipo de enfoque a las relaciones humanas. Quizá la forma más rentable de utilizarlo sería como ayuda para una preparación concienzuda antes de una entrevista, pero entonces el usuario habrá de tener cuidado en no tomarse demasiado al pie de la letra los consejos; al menos no hasta que los ordenadores se conviertan en máquinas realmente pensantes.

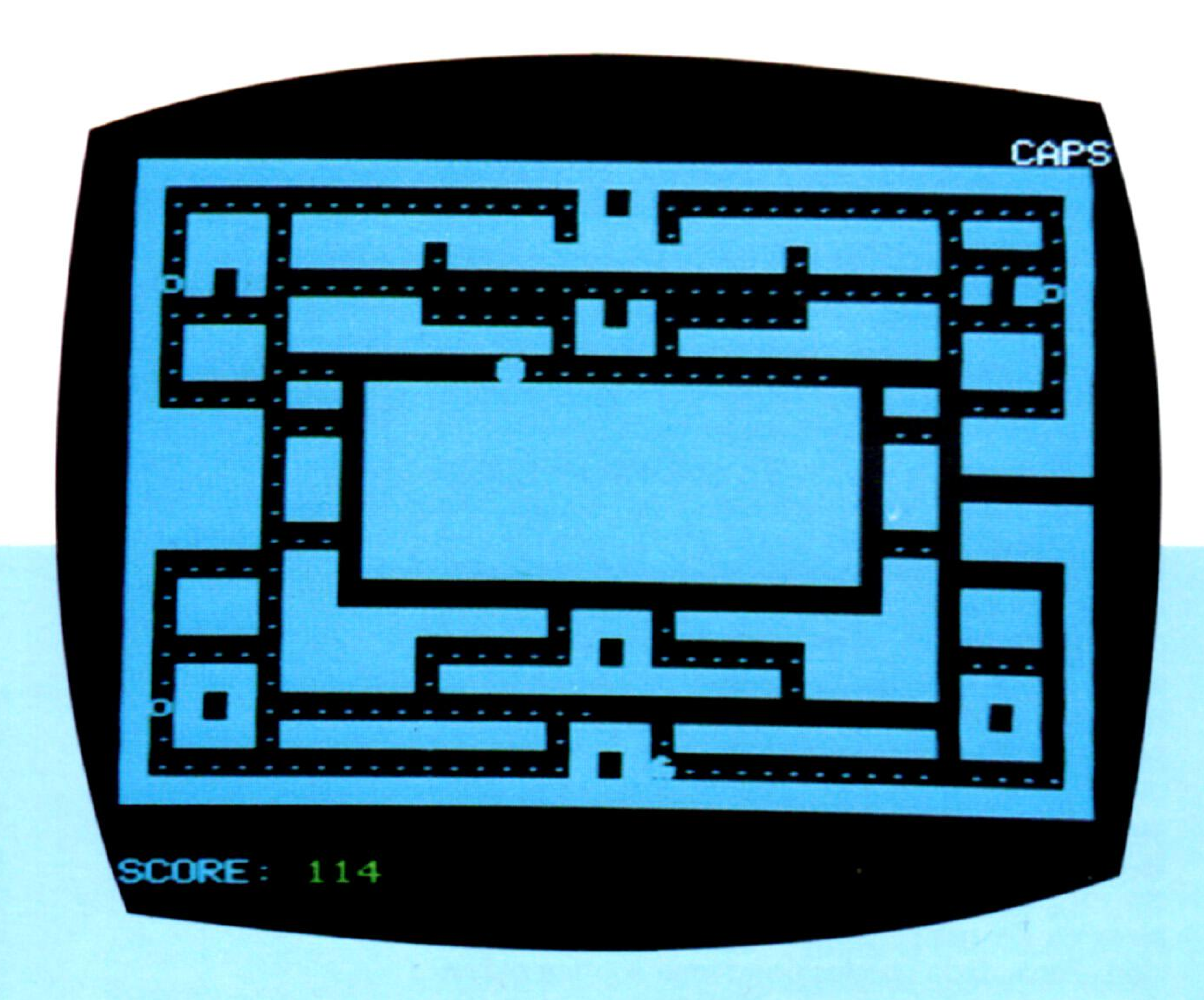


The Human Edge
Juego de cuatro paquetes (que se pueden adquirir por separado) para máquinas MS-DOS y PC-DOS
Distribuidor:
Thorn EMI Computer Software Distributors, 296 Farnborough Road, Farnborough, Hants GU14 7NF, Gran Bretaña
Autores:
Human Edge Software, California
Formato:
Disco

Mind Prober
Para el Commodore 64, el Apple II y el Macintosh
Distribuidor
El mismo
Autores:
Los mismos
Formato:
Disco

Pacman

Presentamos una versión para el Oric de este famoso y pionero juego de laberinto



Este listado ofrece dos particularidades:

1. Hay sólo un fantasma.
2. Al ingerir las "píldoras de energía", el jugador obtiene puntos, pero no puede comerse al fantasma.

Utilice las teclas de control del cursor para desplazarse por la pantalla.

```
30 RE=0:GOSUB 9000
40 GOSUB 8000
50 V%=19:H%=19:GOSUB 7000
60 PLOT H%,V%," "
70 A$=KEY$
80 A=DEEK(783)
90 REM
95 H1%=H%:V1%=V%
100 REM
110 IF A=48351 THEN H%=H%-1:M$="'"
120 IF A=48255 THEN H%=H%+1:M$="%"
130 IF A=48319 THEN V%=V%+1:M$="&"
140 IF A=48375 THEN V%=V%-1:M$="$"
150 IF H%<1 THEN H%"38
160 IF H%>38 THEN H%=1

165 PLAY 0,1,1,10
170 IF SCRN(H%,V%)=44 THEN V%=V1%:H%=H1%
    :GOTO 210
180 IF SCRN(H%,V%)=46 THEN SC%=SC%+1:CO%
    =CO%+1
190 IF SCRN(H%,V%)=111 THEN GOSUB 1000
200 IF SCRN(H%,V%)=43 THEN GOTO 5000
210 PLOT H%,V%,M$
220 PLOT 1,25,"PUNTOS:"+STR$(SC%)
230 PLOT X%,Z%,C$
240 Z1%=Z%:X1%=X%
245 IF M1%>3 THEN M1%=0
250 IF M1%=0 THEN Z%=Z%-1
260 IF M1%=1 THEN X%=X%-1
270 IF M1%=2 THEN Z%=Z%+1
280 IF M1%=3 THEN X%=X%+1
281 PLOT H%,V%,"#"
290 IF SCRN(X%,Z%)=44 THEN Z%=Z1%:X%=X1%
    :M1%=INT(RND(1)*4)
300 IF SCRN(X%,Z%)<40 AND SCRN(X%,Z%)>33
    THEN GOTO 5000
310 C$=CHR$ (SCRN(X%,Z%))
320 PLOT X%,Z%,"+"
330 IF LV%<1 THEN GOTO 2000
340 IF CO%=T% THEN CO%=0:GOTO 50
350 GOTO 60
1000 SC%=SC%+(INT(RND(1)*20))+20
1010 ZAP
1020 RETURN
2000 CLS
2005 ZAP
2010 PRINT,,,,"SE ACABO"
2020 PRINT
2030 PRINT,,,"PUNTOS";SC%
2040 PRINT
2050 PRINT
2060 IF SC%>RE THEN RE=SC%
2070 PRINT,,"MARCADOR ACTUAL:"RE
2080 PRINT
2090 PRINT
2100 PRINT,"PULSAR UNA TECLA PARA JUGAR"
2110 IF KEY$<>"" THEN GOTO 2120
2120 GET K$
2130 GOTO 40
5000 PLOT X%,Z%,"-"
5010 PLOT H%,V%,"#"
5020 PLAY 1,0,1,10
5030 WAIT 15
5040 PLOT H%,V%,"("
5050 PLAY 1,0,1,20
5060 WAIT 20
5070 PLOT H%,V%,")"
5080 PLAY 1,0,1,25
5090 WAIT 30
5100 PLOT H%,V%,"*"
5110 PLAY 1,0,1,30
5120 WAIT 40
5130 PLOT H%,V%," "
5140 PLAY 1,0,1,50
5150 EXPLODE:WAIT 100
5160 CO%=0:V%=12:H%=16:LV%=LV%-1
5170 IF LV%<1 THEN 2000
5180 GOTO 50
7000 CLS:PAPER 0:INK INT (RND(1)*4)+3
7005 PLOT 0,25,6
7006 PING
7010 PRINT",.......................................................
    ........"
7020 PRINT",---------------,  ,---------
    -------,"
7030 PRINT",-...-..............-........
    ,,-,,,-,"
7040 PRINT",-...-.....-..............-...
    ...-----,"
7050 PRINT",0,  ,-------------------------------------------
    ---,  ,0,"
7060 PRINT:,-----,.....------,  ,------,,,
    ,,-----,"
7070 PRINT",-...-..........-...-........
    ,,-,,,-,"
7080 PRINT",-,,, -------------------------------------------
    ---,,,-,"
7090 PRINT",-----,,-.....................-
    ,,-----,"
7100 PRINT",.....----.....................-
    ---,,,,,"
7110 PRINT",.....-,,-.....................-
    ,,-,,,,,"
7120 PRINT",.....-,,-.....................-
    ,,------"
7130 PRINT",.....-,,-.....................-
    ,,-,,,,,"
7140 PRINT",.....----.....................-
    ---,,,,,"
7150 PRINT",-----,,-.....................-
    ,,-----,"
7160 PRINT",-,,,-,, -------------------------------------------
    ,,-,,,-,"
7170 PRINT",-...-..........-...-........
    ,,-,,,-,"
7180 PRINT",-----,.....------,  ,------,,,
    ,,-----,"
7190 PRINT",-...-.....-..............-...
    ,,-,,,-,"
7200 PRINT",o,  ,-------------------------------------------
    ---,  ,0,"
7210 PRINT",-...-..........-...-........
    ,,-,,,-,"
7220 PRINT",---------------,  ,---------
    -------,"
7230 PRINT".......................................................
    ........"
7500 RETURN
8000 M$="%"
8010 C1%=8:M1%=0
8020 SC%=0
8030 Z%=7:X%=19
8040 LV%=3
8050 CO%=0
8060 T%=312
8070 RETURN
9000 FOR U=(46080+(ASC("#")*8)) TO (46080
    +(ASC(",")*8)+7)
9010 READ US:POKE U,US:NEXT U:RETURN
9020 DATA 0,30,63,63,63,63,63,30
9030 DATA 0,18,51,51,63,63,30,12
9040 DATA 0,30,63,60,56,60,63,30
9050 DATA 0,12,30,63,63,51,51,18
9060 DATA 0,30,63,15,7,15,63,30
9070 DATA 0,0,0,0,63,63,63,30
9080 DATA 0,0,0,0,8,12,12,30
9090 DATA 0,33,18,12,0,12,18,33
9100 DATA 0,12,30,56,63,63,63,42
9110 DATA 63,63,63,63,63,63,63,63
```



Mejorando lo presente

Acaba de aparecer el RS128, de Memotech, que incorpora facilidades completas de interface

A pesar de ser máquinas tenidas en muy buena estima, la serie de microordenadores Memotech 500 (el MTX500 y MTX512) ha sido ignorada en gran medida por los compradores de ordenadores personales. Características atractivas (tales como gráficos en alta resolución, un ensamblador incorporado, un BASIC sofisticado y un lenguaje exclusivo para tratamiento de textos denominado NODDY) por cierto no han restado mérito a estas máquinas; el fracaso que representa no haber conseguido un gran éxito a nivel popular se puede atribuir a que quedan comprendidas entre dos segmentos diferentes del mercado de ordenadores personales.

Por un lado, su precio tal vez las haga poco asequibles al amante de los juegos, quien puede pensar que las características más sofisticadas no valen el costo más elevado. Por otra parte, el usuario "serio" (Memotech afirma que la serie está dirigida al usuario de pequeña empresa) puede desanimarse por el hecho de que la serie 500 carezca de interfaces incorporadas, lo que permitiría la conexión de unidades de disco. Estas interfaces sí se pusieron a la venta, pero salieron como elementos separados, diseñados para instalarse en un conector marginal situado en el interior de las máquinas. Esto no es demasiado sorprendente tratándose de una empresa que cimentó su nombre a través de la producción de placas accesorias para el ZX81, pero si es posible que haya fracasado en impresionar a los usuarios que deseaban una máquina lista para enchufar y poner en funcionamiento. Memotech parece haber reconocido este problema y ha introducido el RS128, una máquina con interfaces incorporadas.

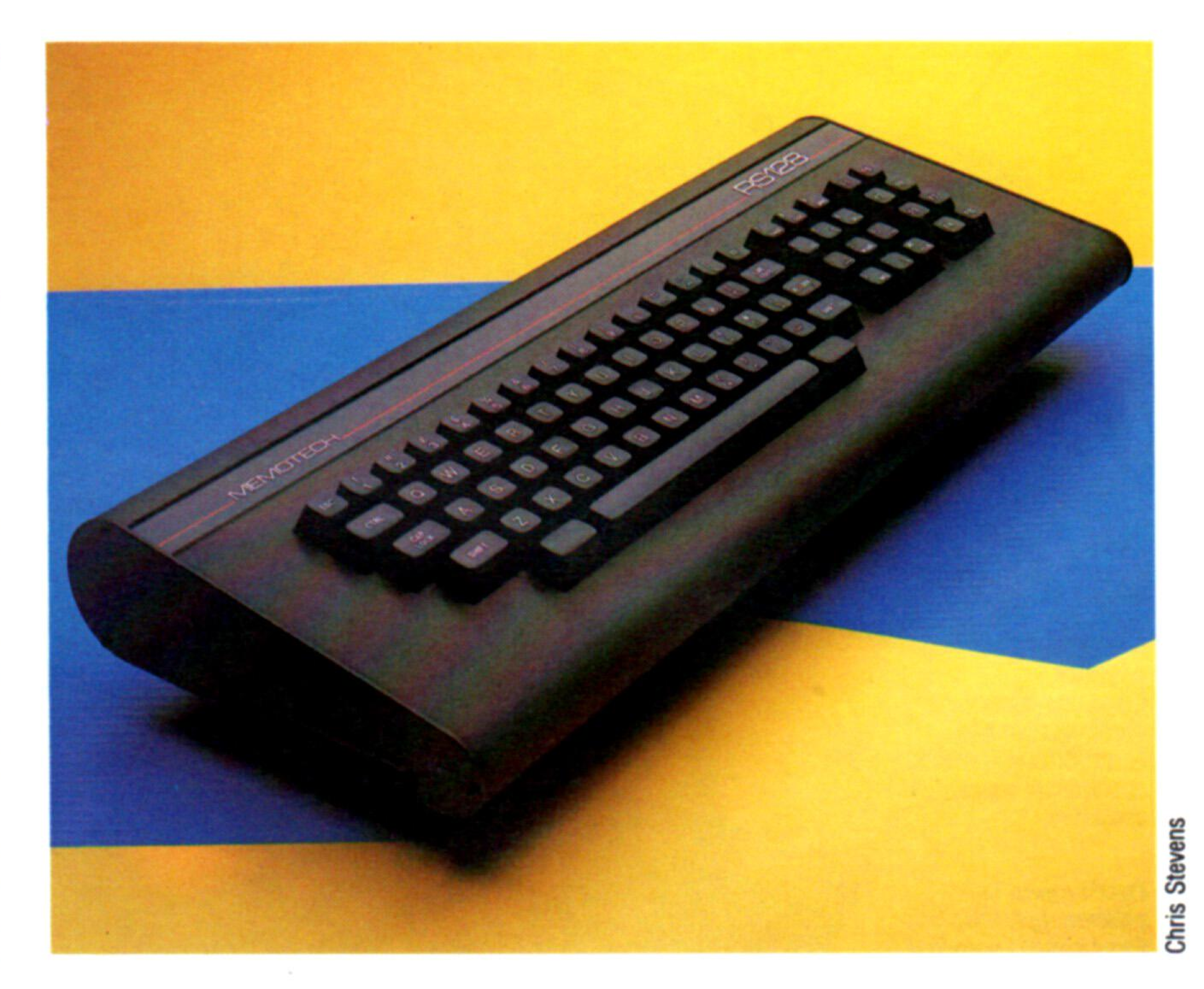
El aspecto de la máquina

A primera vista, el RS128 parece idéntico a la serie 500; da la impresión de ser elegante y estar un poco por encima del nivel medio del mercado. Al igual que sus medio hermanas, la máquina tiene una carcasa de aluminio en vez de la habitual de plástico, y ello hace que la máquina Memotech sea considerablemente más pesada que la mayoría del resto de micros. Hay un teclado QWERTY estándar y un teclado numérico, que lleva algunas de las instrucciones para el lenguaje de programación de textos NODDY. A la derecha del teclado numérico hay, asimismo, ocho teclas de función programables. Las teclas poseen un tacto excelente y están diseñadas de acuerdo a un elevado estándar profesional.

El trazado tiene algunos problemas menores: la tecla Return no es mucho más grande que las teclas normales y al principio los mecanógrafos al tacto tendrán dificultades para localizarla, y la tecla Delete no está en el teclado de máquina de escribir propiamente dicho, sino situada en la zona del teclado numérico. En el rincón superior derecho hay una tecla Backspace pero, a diferencia de la mayor parte de los teclados de ordenador, en los que Backspace actúa también como tecla Delete-left (conocida como "retroceso con eliminación"), en el Memotech es simplemente un cursor-izquierda.

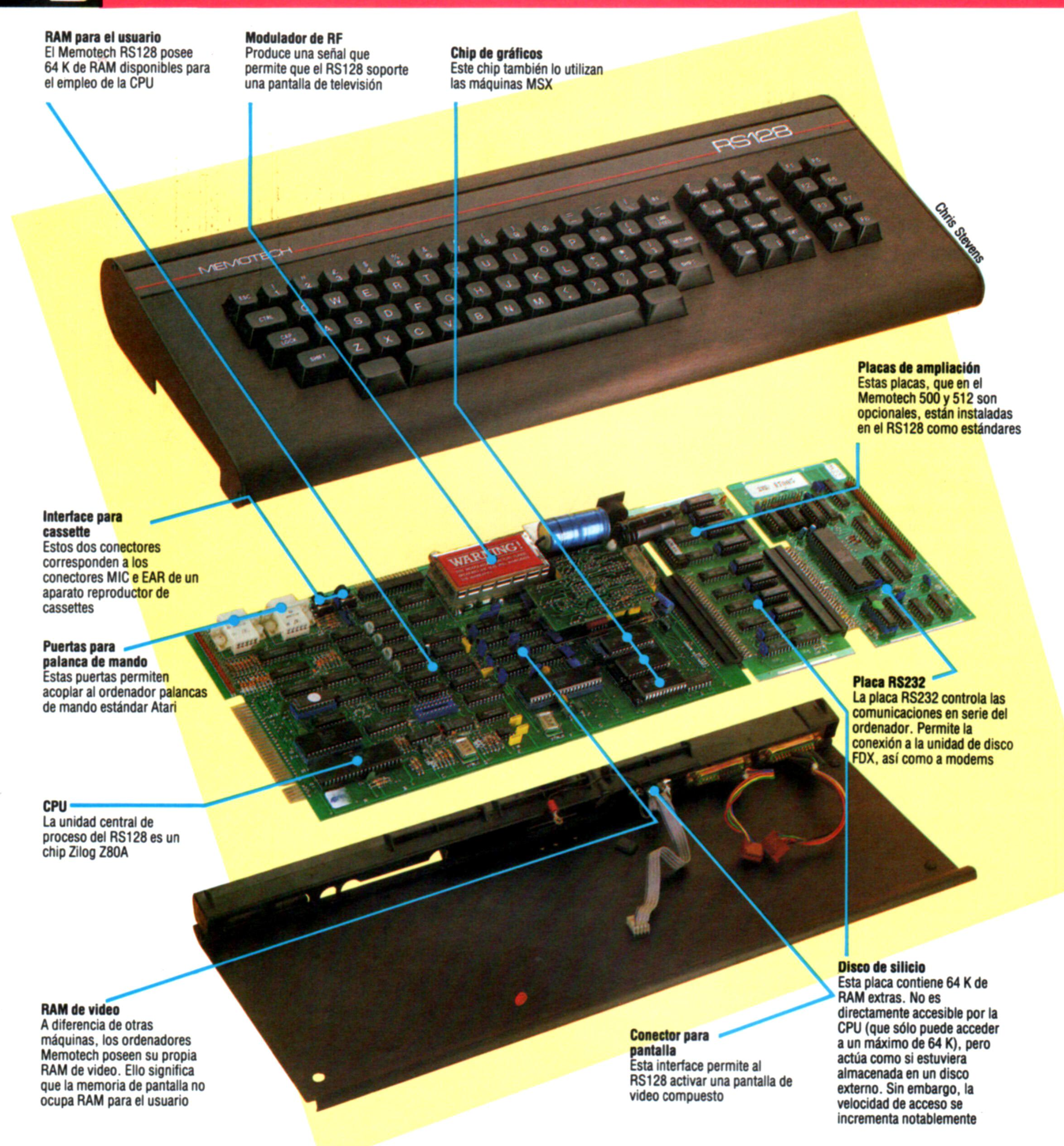
En la parte posterior de la máquina hay numerosas interfaces. Algunas de éstas se suministraban con la serie 500 y otras son adiciones recientes. En el extremo izquierdo hay un par de puertas RS232 que permiten conectar la máquina a unidades de disco flexible FDX. Estas puertas también se pueden utilizar para otros fines, como impresoras en serie y comunicaciones en red. A la derecha de las RS232 hay un enchufe hembra para video compuesto y un enchufe hembra para alta fidelidad (este último permite amplificar el sonido del ordenador a través de un sistema estéreo normal). El conector de potencia y el enchufe hembra RF vienen a continuación, seguidos por una interface para impresora tipo Centronics. La interface para cassette se compone de un par de conectores *microjack*, para EAR y MIC, en el mismo estilo que el Sinclair Spectrum. Por último, hay un par de puertas para palanca de mando tipo Atari de nueve patillas.

Las puertas para interfaces están señaladas con letras blancas, que se pueden leer claramente desde la parte de atrás de la máquina. Esto parece que podría permitir enchufar los periféricos sin tener



Chris Stevens

Modelo perfeccionado
El Memotech RS128 es una versión perfeccionada de la serie MTX500. Este nuevo modelo está dotado de conectores RS232 gemelos, lo que permite que la máquina soporte las unidades de disco flexible FDX. Ello significa que el ordenador es particularmente atractivo para el usuario serio de un micro personal o para el usuario de una pequeña empresa



RAM para el usuario
El Memotech RS128 posee 64 K de RAM disponibles para el empleo de la CPU

Modulador de RF
Produce una señal que permite que el RS128 soporte una pantalla de televisión

Chip de gráficos
Este chip también lo utilizan las máquinas MSX

Placas de ampliación
Estas placas, que en el Memotech 500 y 512 son opcionales, están instaladas en el RS128 como estándares

Interface para cassette
Estos dos conectores corresponden a los conectores MIC e EAR de un aparato reproductor de cassettes

Puertas para palanca de mando
Estas puertas permiten acoplar al ordenador palancas de mando estándar Atari

Placa RS232
La placa RS232 controla las comunicaciones en serie del ordenador. Permite la conexión a la unidad de disco FDX, así como a modems

CPU
La unidad central de proceso del RS128 es un chip Zilog Z80A

Disco de silicio
Esta placa contiene 64 K de RAM extras. No es directamente accesible por la CPU (que sólo puede acceder a un máximo de 64 K), pero actúa como si estuviera almacenada en un disco externo. Sin embargo, la velocidad de acceso se incrementa notablemente

RAM de video
A diferencia de otras máquinas, los ordenadores Memotech poseen su propia RAM de video. Ello significa que la memoria de pantalla no ocupa RAM para el usuario

Conector para pantalla
Esta interface permite al RS128 activar una pantalla de video compuesto

que inclinarse por arriba para mirar la parte de atrás. Lamentablemente, Memotech ha colocado las puertas en depresiones de la máquina, de modo que el usuario tendrá igualmente que estirarse por arriba para localizar los enchufes.

La pantalla de BASIC, de 24 por 40 caracteres, está dividida en tres secciones, que se aprecian en el momento del encendido. Las 19 filas superiores corresponden a la pantalla principal, donde se visualizan los listados de programas. Debajo está la pantalla EDIT, donde se entran las nuevas líneas. En la parte inferior de la pantalla hay una única línea para la visualización de mensajes de error. Al igual que las máquinas Sinclair, las líneas de programa se modifican mediante el empleo de una instrucción EDIT. Además, el sistema operativo no permite la inserción de una línea en el programa desde la pantalla EDIT si la línea contiene un error de sintaxis.

El BASIC es muy similar al MSX, con instrucciones tales como SOUND, PAPER, INK y CIRCLE. Sin

mbargo también contiene algunas útiles instrucciones de las que carece el BASIC MSX. Éstas, en onjunto, están relacionadas con las capacidades ara tratamiento de pantalla de la máquina. A odo de ejemplo, la instrucción CSRx,y posicionará l cursor en el punto de la pantalla especificado por s coordenadas (x,y). Una instrucción más poten- es es CRVS, que permite que el usuario defina una entana en cualquier lugar de la pantalla. Dentro e estas ventanas se puede visualizar tanto texto omo gráficos.

El lenguaje tiene incorporadas, asimismo, ins- rucciones para controlar sprites. En este sentido, na instrucción particularmente útil es GENPAT, que ermite establecer el patrón del sprite, en vez de ener que colocar el patrón en sentencias de datos. os gráficos del Memotech los proporciona el chip e video TMS9929A, que es el especificado para s máquinas MSX.

El procesador central de las máquinas Memotech s el Z80 y éste, por supuesto, les permite ejecutar l sistema operativo CP/M. Muchos pequeños fa- ricantes de ordenadores eligieron el procesador Z80 porque ejecuta CP/M, lo que evita el problema e tener que generar una gran base de software ara que los usuarios puedan sacar el máximo parti- o de un ordenador nuevo. Evidentemente, para provechar al máximo el CP/M, el ordenador ha de ener una pantalla de 80 columnas y, aunque es inu- ual, Memotech proporciona, dentro de la unidad e disco, una placa para uso de 80 columnas. Las nidades, por su parte, son de doble cara y doble ensidad, y su velocidad de transferencia al ordena- or es de 9 200 baudios.

Con la unidad de disco se suministra un lote de oftware. Aparte del disco de sistema CP/M, el pa- uete incluye el procesador de textos *NewWord*, la oja electrónica *SuperCalc, Compact* y *Televideo*, ue permiten que las unidades lean discos escritos n otros formatos de disco (según Memotech, ello ncluye a los discos IBM) y *Contact*, que hace posi- le enlazar la segunda puerta RS232 en un sistema e red.

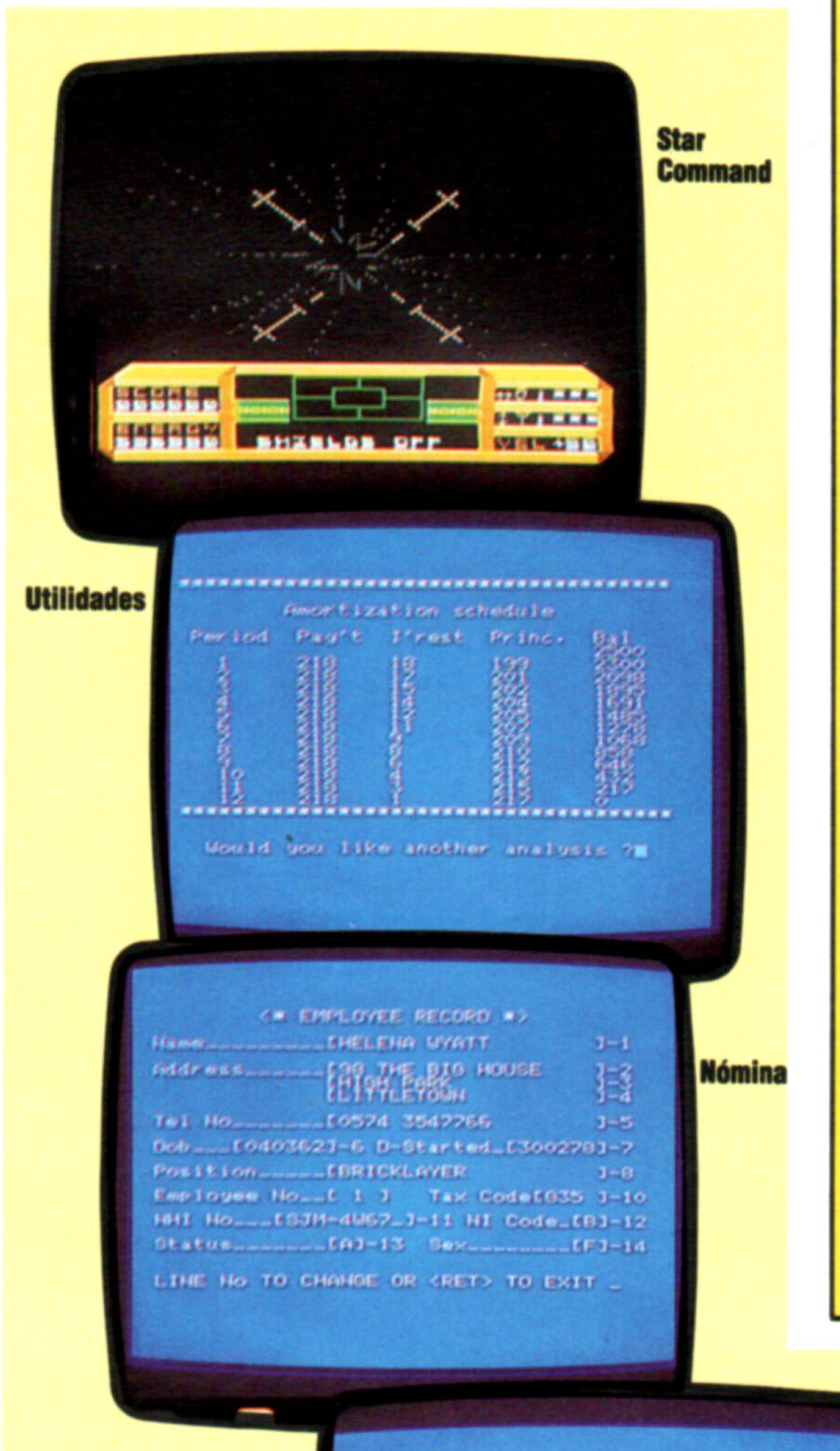
El RS128 posee 128 K de RAM. No obstante, ado que emplea un procesador de 8 bits, sólo es apaz de direccionar 64 K, utilizándose los otros 64 K como un "disco de silicio". Un disco de silicio lmacena archivos y programas exactamente de la nisma forma en que lo hace un disco flexible, pero omo está retenido en chips, es hasta 50 veces más ápido que un disco flexible convencional. La infor- nación almacenada en un disco de silicio se trans- iere a RAM direccionable cuando así se requiere.

El manual que se proporciona con la máquina es nucho más grande que los que se suministran nor- nalmente con ordenadores personales, si bien ello e debe básicamente a que no ha sido tipografiado, no a que incluya mucha más información. Sin em- argo, Memotech ha incluido todos los detalles téc- icos que puede necesitar un usuario, incluyendo iagramas de circuitos y llamadas al sistema opera- ivo.

Al perfeccionar la serie 500 con el RS128, la em- resa se ha esforzado tenazmente por producir un rdenador de oficina estándar. Por su precio, la náquina ciertamente entra en competencia con el inclair QL, el BBC Micro y el Commodore Plus/4. in embargo, queda por ver si generará un volu- nen de ventas comparable al de sus rivales.



Unidad de disco gemela FDX
Esta unidad permite que el ordenador ejecute el sistema operativo CP/M. Cada una de las unidades de disco de 5 1/4 pulgadas puede almacenar hasta 500 K de información



MEMOTECH RS128

DIMENSIONES
488×202×56 mm

CPU
Z80A operando a 4 MHz

MEMORIA
64 K de RAM y 24 K de ROM

PANTALLA
40×24 en modalidad de textos. Modalidad de texto con gráficos: texto en 32×24 y 256×192 pixels en 16 colores. Existen asimismo facilidades para controlar hasta 32 sprites de forma independiente

INTERFACES
Puertas para cassette (MIC e EAR); interface de E/S; dos puertas para palanca de mando; dos puertas RS232; enchufe para hi-fi; enchufe para video compuesto; enchufe para TV; interface en paralelo

LENGUAJES
BASIC, FORTH, PASCAL

TECLADO
57 teclas tipo máquina de escribir; las teclas F y J están ahuecadas para localización con el dedo. Doce teclas de función en un teclado numérico y ocho teclas de función programables

DOCUMENTACION
El manual que se entrega es el mismo que el de la serie MTX500

VENTAJAS
La adición de las placas RS232 hacen que el RS128 sea una adquisición muy atractiva para el usuario personal "serio" y los usuarios de gestión de pequeñas empresas

DESVENTAJAS
Hay poco software escrito especialmente para la máquina

Software de apoyo

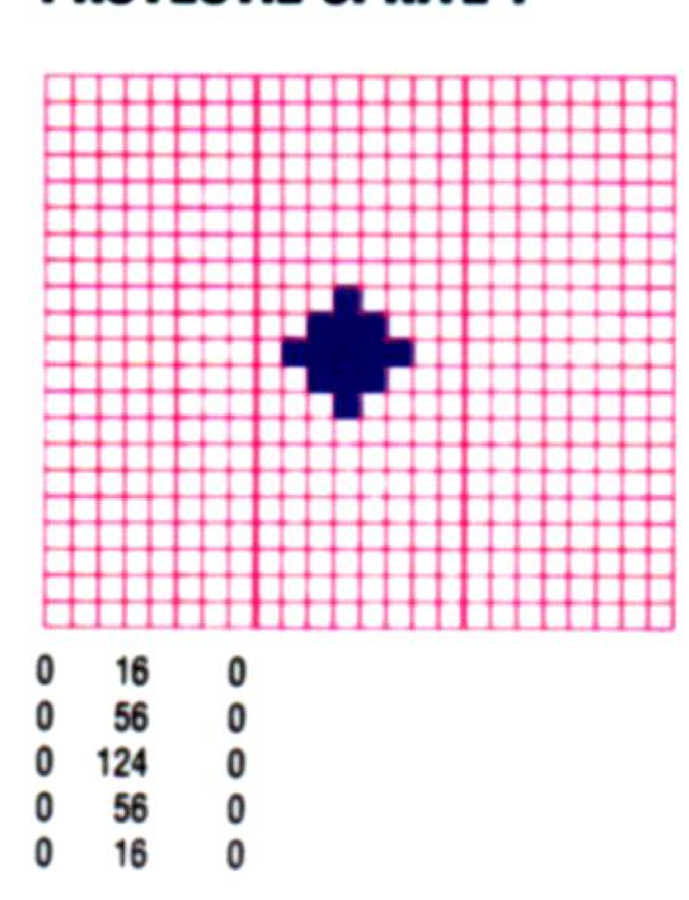
Éstos son algunos de los paquetes de gestión y de juegos que se venden para las máquinas Memotech. En comparación con otras máquinas, el software disponible es bastante limitado, pero el acceso a la bolsa común de software CPM solucionará en gran medida este problema

Alta resolución

Ahora diseñaremos y programaremos las visualizaciones de escenarios especiales para el Commodore 64

¡Dispárales!
La acción de "disparo" de la puerta para palanca de mando se consigue en el 64 mediante el empleo de un sprite definido como proyectil. Los contornos se consiguen colocando (POKE) ceros en la zona de definición, leyendo (READ) luego las zonas sólidas de sentencias DATA y colocándolas en las posiciones de sprites adecuadas

PROYECTIL-SPRITE 1



Estiramiento de sprites
Los valores listados para el sprite de la puerta para palanca de mando se leen de sentencias DATA y se colocan en las posiciones de sprites apropiadas. Tal como está diseñada, la puerta para palanca de mando está demasiado comprimida, pero se la puede ensanchar horizontalmente (como vemos en la ilustración) cambiando el valor del registro de ampliación horizontal

El Commodore 64 dispone de facilidades para alta resolución, pero éstas sólo están disponibles para el programador en lenguaje máquina, ya que no se facilita ninguna instrucción en BASIC para el tratamiento de alta resolución, y llevar a cabo los PEEK y POKE correspondientes en BASIC para producir visualizaciones en alta resolución es tan lento que esta aplicación resulta inoperante. En cambio, hemos de adaptar las facilidades, relativamente fáciles de utilizar, que ofrece la máquina.

Se pueden emplear caracteres para gráficos para construir letras grandes u otras visualizaciones mediante las combinación de diferentes caracteres. Los sprites son un método conveniente de introducir formas en alta resolución en la pantalla normal del Commodore. Los caracteres PET se pueden posicionar en la pantalla utilizando ya sea una serie de sentencias PRINT o bien colocando (POKE) el código de carácter correspondiente en la pantalla. Vamos a demostrar ambos procedimientos.

Las líneas 8020-8170 del listado de la pantalla de la palanca de mando comprenden la lectura de los datos para los dos sprites utilizados en esta rutina. El primero, el sprite 0, se define mediante los primeros 63 números del grupo de sentencias DATA comprendidas entre las líneas 8450 y 8497, y representa la puerta para palanca de mando (ilustrada en el diagrama). Los datos de sprites por lo general se sitúan en la parte superior de la zona para programas en BASIC, pero cuando se trata de un programa en BASIC extenso estos datos tienen grandes posibilidades de ser machacados. Una ubicación alternativa es la zona del buffer de la cassette, entre las posiciones 832 y 1022, donde se puede almacenar los datos de tres sprites. Esta rutina almacena sus datos de sprites en esta zona segura.

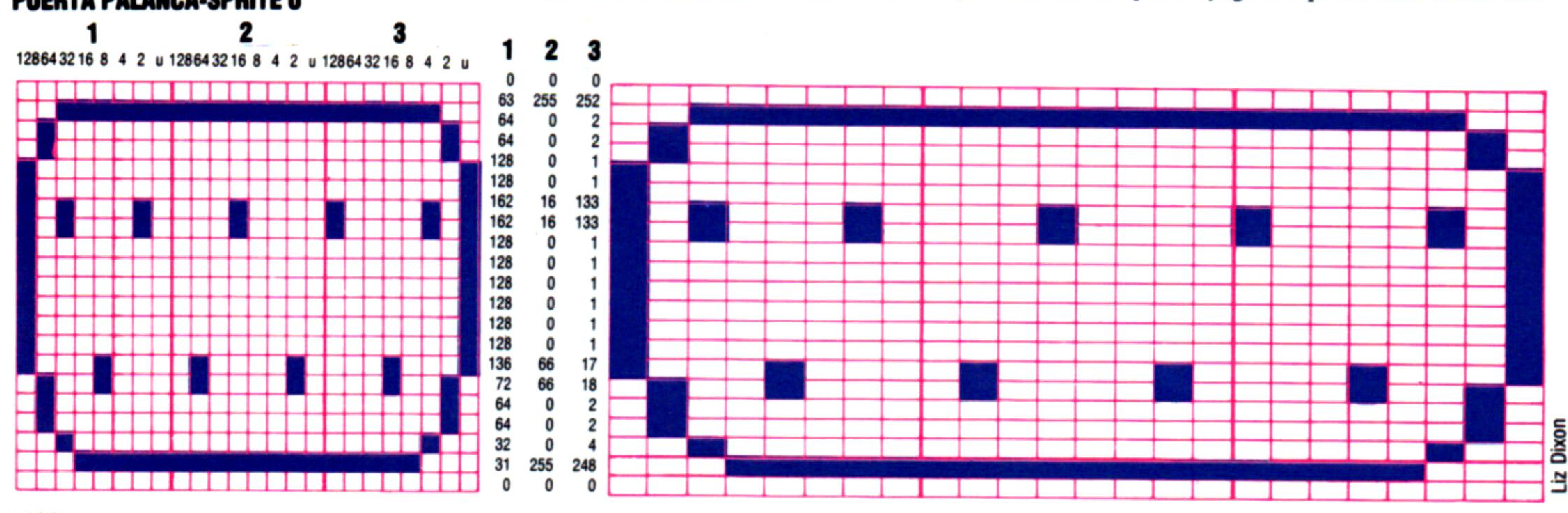
El sprite 0 sufre una ampliación al doble de su anchura original, para producir la forma final visualizada, colocando a uno el bit 0 del registro de ampliación horizontal en la línea 8170. Observe que todos los registros que controlan los atributos de los sprites, como color, posición y ampliación, están relacionados con la dirección de comienzo para el chip controlador de video (VIC). Recordar que el registro de ampliación horizontal tiene la dirección VIC+29 es mucho más fácil que memorizar su verdadera posición de memoria (53277). Algunos atributos de los sprites requieren un registro entero para cada sprite (p. ej., los registros de las coordenadas X e Y); pero cuando los ocho bits controlan independientemente una función para los ochos sprites disponibles, los atributos se pueden controlar activando y desactivando el bit apropiado en un único registro. El sprite 1 se define mediante los 13 números restantes de las sentencias DATA y representa un objeto a disparar desde la puerta para la palanca de mando.

Dado que la parte "sólida" del sprite 1 es pequeña (representa un proyectil), es más rápido y sencillo entrar en dos etapas los 63 bytes de datos que lo definen. Primero, colocando (POKE) 63 ceros en la zona de definición, y luego leyendo (READ) y colocando (POKE) los pocos números que definen la forma. De esta manera podemos omitir la gran cantidad de ceros que, de lo contrario, se requerirían como datos.

Las líneas 8190-8220 se refieren a la construcción de series compuestas por un conjunto de caracteres para gráficos PET. LE$ forma una línea horizontal en toda la anchura de la pantalla, mediante la combinación de 40 caracteres PET especiales de la derecha de la tecla C. DW$ es una serie de caracteres de cursor-abajo. LS$ y RS$ son grupos de diagonales a izquierda y derecha (en frente y a la derecha de las teclas N y M) que se utilizan para formar un patrón de espina de pescado en el primer plano. Este patrón crea en la escena una sensación de profundidad y perspectiva.

La rutina "Disparo" de la línea 8310 elige un punto al azar de la parte inferior de la pantalla y dirige el sprite 1 hacia el mismo, repitiéndose el proceso hasta que el jugador pulsa una tecla. Los

PUERTA PALANCA-SPRITE 0



colores de la pantalla se restauran a los normales, se limpia la pantalla y se apagan los sprites antes de retornar al programa principal. Para utilizar esta subrutina con *Digitaya*, debe insertarse esta línea:

```
3845 GOSUB 8000: REM IMAGEN DE LA PUERTA
     PARA PALANCA DE MANDO
```

El otro listado proporciona una visualización gráfica para el escenario ALU de *Digitaya* y demuestra distintos métodos de visualizar caracteres en la pantalla. Las líneas de la 7040 y la 7090 leen un cierto número de sentencias DATA y colocan (POKE) los valores directamente en la zona de pantalla. En la posición correspondiente de la zona de color también se le coloca (POKE) el código de color para el carácter. En este ejemplo el código de color es 2, haciendo que los caracteres se visualicen en rojo.

Para conseguir que las grandes letras ALU se desplacen en la pantalla se aplica un truco bastante inusual. La primera línea de códigos de caracteres gráficos que han de conformar las letras ALU se colocan (POKE) en la segunda línea de la pantalla. Entonces se llama a la subrutina de la línea 7680, haciendo que la pantalla se desplace una línea hacia abajo. Luego se puede colocar (POKE) la segunda línea de códigos en la misma zona de pantalla que la primera, y llamar nuevamente a la subrutina. La repetición de este proceso para cada una de las ocho líneas de código hace que las letras ALU parezcan desplazarse desde la parte superior de la pantalla.

Se demuestran otros dos procedimientos para presentar los datos de caracteres en la pantalla. Los caracteres se pueden imprimir (PRINT) directamente, como en las líneas 7130 y 7140, o leer como una serie de datos a imprimir, como es el caso del diseño del signo de interrogación en las líneas 7170 y 7590-7670. Este segundo método permite una sencillez de diseño dentro de las sentencias DATA.

Para utilizar esta rutina añada la siguiente línea:

```
4565 GOSUB 7000:REM IMAGEN ALU
```

Escritura de letras
El escenario de la ALU para *Digitaya* se crea a partir de tres caracteres para gráficos PET en baja resolución, como se puede apreciar. Las grandes letras formadas parecen desplazarse hacia abajo, desde la parte superior de la pantalla hasta su posición de descanso

Carácter para gráficos PET	Códigos de pantalla Normal	Invertido
	32	160
	105	233
	95	223

Liz Dixon

Pantalla ALU

```
7000 REM **** S/R IMAGEN ALU ****
7010 VIC=53248:CS=55296:SC=1024
7020 PRINT CHR$(147):REM LIMPIAR PANTALLA
7030 POKE VIC+32,0:POKE VIC+33,0:REM ESTABLECER PANTALLA/BORDE
7040 CC=0
7050 FOR J=1 TO 8:GOSUB 7680:REM DESPLAZ.
7060 FOR I=47 TO 72
7070 READ A:CC=CC+A:POKE SC+I,A:POKE CS+I,2
7080 NEXT I,J
7090 READ CS:IF CS<>CC THEN PRINT"ERROR SUMA DE CONTROL":STOP
7100 GOSUB 7680:REM DESPLAZ.
7110 PRINTCHR$(158):REM TEXTO AMARILLO
7120 FOR I=1 TO 8:PRINT:NEXT I:REM MOVER HACIA ABAJO
7130 PRINTTAB(9)"AND";SPC(7);"OR";SPC(8);"NOT"
7140 PRINTTAB(10)"0";SPC(9);"0"SPC(9);"0"
7150 PRINTCHR$(28):REM PRUEBA ROJO
7160 REM ** SIGNO INTERROGACION **
7170 FOR I=1 TO 9:READ Q$:PRINTTAB(16)Q$:NEXT I
7180 REM **** ESPERAR TECLA Y REINICIALIZAR ****
7190 GET A$:IF A$="" THEN 7190
7200 POKE VIC+32,14:POKE VIC+33,6:REM PANTALLA/BORDE
7210 PRINTCHR$(154): REM LT TEXTO AZUL
7220 PRINTCHR$(147):REM LIMPIAR PANTALLA
7230 RETURN
7240 REM **** DATOS PANTALLA ****
7250 REM ** FILA 1 **
7260 DATA 160,32,32,32,32,160,32,32,32,32
7270 DATA 95,160,160,160,160,105
7280 DATA 32,32,32,32,95,160,160,160,160,105
7290 REM ** FILA 2 **
7300 DATA 160,32,32,32,32,160,32,32,32,32
7310 DATA 160,223,32,32,32,32,32,32,32,32
7320 DATA 160,223,32,32,233,160
7330 REM ** FILA 3 **
7340 DATA 160,32,32,32,32,160,32,32,32,32
7350 DATA 160,32,32,32,32,32,32,32,32,32
7360 DATA 160,32,32,32,32,160
7370 REM ** FILA 4 **
7380 DATA 160,160,160,160,160,160,32,32,32,32
7390 DATA 160,32,32,32,32,32,32,32,32,32
7400 DATA 160,32,32,32,32,160
7410 REM ** FILA 5 **
7420 DATA 160,32,32,32,32,160,32,32,32,32
7430 DATA 160,32,32,32,32,32,32,32,32,32
7440 DATA 160,32,32,32,32,160
7450 REM ** FILA 6 **
7460 DATA 233,105,32,32,95,223,32,32,32,32
7470 DATA 160,32,32,32,32,32,32,32,32,32
7480 DATA 160,32,32,32,32,160
7490 REM ** FILA 7 **
7500 DATA 32,233,105,95,223,32,32,32,32,32
7510 DATA 160,32,32,32,32,32,32,32,32,32
7520 DATA 160,32,32,32,32,160
7530 REM ** FILA 8 **
7540 DATA 32,32,233,223,32,32,32,32,32,32
7550 DATA 160,32,32,32,32,32,32,32,32,32
7560 DATA 160,32,32,32,32,160
7570 DATA 14463: REM SUMA CONTROL
7580 REM ** DATOS SIGNO INTERROGACION **
7590 DATA "  ????  "
7600 DATA "?      ?"
7610 DATA "?      ?"
7620 DATA "      ??"
7630 DATA "   ?   "
7640 DATA "   ?   "
7650 DATA "   ?   "
7660 DATA "       "
7670 DATA "   •   "
7680 REM **** S/R DESPLAZAR PANTALLA ****
7690 POKE 218,160
7700 PRINTCHR$(19);CHR$(17);CHR$(157);CHR$(148)
7710 RETURN
```

Pant. puerta palanca

```
8000 REM **** S/R IMAGEN PUERTA PALANCA MANDO ****
8010 PRINTCHR$(147):REM LIMPIAR PANTALLA
8020 S0=832:S1=S0+64:REM DIR COMIENZO DATOS SPRITE
8030 CC=0:VIC=53248:REM COMIENZO DE CHIP VIC
8040 FOR I=S0 TO S0+62:READ A:CC=CC+A:POKE I,A:NEXT I
8050 FOR I=S1 TO S1+62:POKE I,0:NEXT I
8060 FOR I=S1+25 TO S1+37:READ A:CC=CC+A:POKE I,A:NEXT I
8065 READ CS:IF CS<>CC THEN PRINT"ERROR SUMA DE CONTROL":STOP
8070 POKEVIC+33,0:POKEVIC+32,0:REM ESTABLECER PANTALLA/BORDE
8080 REM ** ESTABLECER PUNTEROS SPRITES **
8090 POKE 2040,S0/64:POKE 2041,S1/64
8100 REM ** ESTABLECER REGS CONTROL SPRITES VIC **
8110 POKE VIC+39,7:REM ESTABLECER COLOR SPRITE 0
8120 POKE VIC+40,7:REM ESTABLECER COLOR SPRITE 1
8130 POKE VIC,65:REM ESTABLECER COORD X SPRITE 0
8140 POKE VIC+1,70:REM ESTABLECER COORD Y SPRITE 0
8150 POKE VIC+2,74:REM ESTABLECER COORD X SPRITE 1
8160 POKE VIC+3,70:REM ESTABLECER COORD Y SPRITE 1
8170 POKE VIC+29,1:REM AMPLIAR HORIZ SPRITE 0
8190 LE$="":FOR I=1 TO 40:LE$=LE$+CHR$(195):NEXT I
8200 DW$="":FOR I=1 TO 25:DW$=DW$+CHR$(17):NEXT I
8210 LS$="":FOR I=1 TO 11:LS$=LS$+CHR$(206)+"  ":NEXT I
8220 RS$="":FOR I=1 TO 11:RS$=RS$+CHR$(205)+"  ":NEXT I
8230 PRINTCHR$(158):REM TEXTO AMARILLO
8240 PRINTCHR$(19):PRINT TAB(2)"PUERTA PALANCA MANDO"
8250 PRINTCHR$(154):REM TEXTO AZUL CLARO
8260 PRINTCHR$(19);LEFT$(DW$,17);LE$
8270 PRINT CHR$(145);
8280 PT$=LEFT$(LS$,19)+LEFT$(RS$,21)
8290 PT$=PT$+RIGHT$(LS$,19)+RIGHT$(RS$,21)
8300 FOR I=1 TO 3:PRINT PT$;:NEXT I
8305 POKE VIC+21,3:REM ENCENDER SPRITES 0 & 1
8310 REM ** DISPARAR **
8320 YB=240:Y1=70
8330 X1=74:X=INT(RND(1)*150)+24
8340 G=2*(X-X1)/(YB-Y1)
8350 FOR Y=Y1 TO YB STEP 2
8360 X1=X1+G:POKE VIC+2,X1:POKE VIC+3,Y
8370 NEXT Y
8380 GET A$:IF A$="" THEN 8330
8390 REM ** RESTAURAR PANTALLA **
8400 POKE VIC+21,0:REM APAGAR SPRITES
8410 POKE VIC+32,14:POKEVIC+33,6:REM RESTAURAR PANTALLA/BORDE
8420 PRINT CHR$(147):REM LIMPIAR PANTALLA
8430 RETURN
8440 REM **** DATOS SPRITE ****
8450 DATA 0,0,0,63,255,252,64,0,2,64,0,2
8460 DATA 128,0,1,128,0,1,162,16,133,162,16,133
8470 DATA 128,0,1,128,0,1,128,0,1,128,0,1
8480 DATA 129,0,1,128,0,1,136,66,17,72,66,18
8490 DATA 64,0,2,64,0,2,32,0,4,31,255,248
8495 DATA 0,0,0,16,0,0,56,0,0,124,0,0,56,0,0,16
8497 DATA 3701:REM SUMA DE CONTROL
```

Calcular factoriales

El LOGO es un lenguaje ideal para explorar las matemáticas. A partir de procedimientos sencillos se van realizando tareas más complejas

¿De cuántas formas puede uno acomodar a cuatro personas en cuatro sillas alrededor de una mesa? La primera persona se puede sentar en cualquiera de las cuatro sillas, pero después de haberse sentado, para la segunda persona sólo quedan tres opciones, luego quedan dos para la tercera, y a la última persona sólo le queda un lugar. Por lo tanto, la cantidad total de arreglos diferentes es 4×3×2×1. Esto por lo general se escribe como 4! y se lee "4 factorial". Los factoriales se encuentran con frecuencia en problemas matemáticos que entrañen variaciones, combinaciones y probabilidades.

Es fácil escribir una definición recursiva para calcular factoriales. En primer lugar, debemos observar que el factorial de 0 se define como 1. El factorial de cualquier número positivo distinto de cero (p. ej., x) es el factorial de $x-1$ multiplicado por x. Traduciendo esto en un programa obtenemos:

```
TO FACTORIAL :X
  IF :X=0 THEN OUTPUT 1
  OUTPUT (FACTORIAL :X–1)*:X
END
```

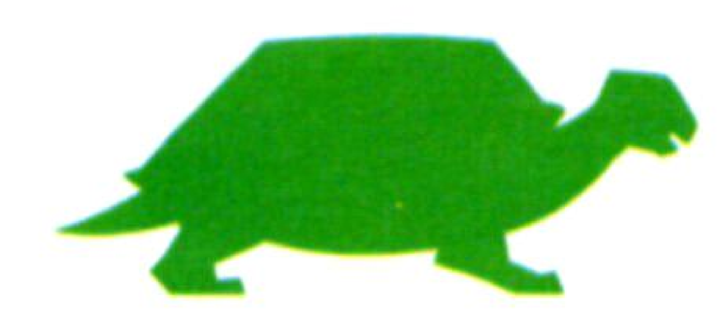
Para probarlo, digite PRINT FACTORIAL 6; el resultado será 720.

Este procedimiento funciona bien hasta el número 12, pero, a partir de éste, los números se vuelven demasiado grandes como para que el ordenador los pueda retener como enteros. En el Commodore 64, por ejemplo, PRINT FACTORIAL 13 dio 6.22702E9, es decir, 6,22702 veces 10^9. Esto es poco satisfactorio, puesto que se han perdido los cuatro últimos dígitos. Existen muchas razones (incluyendo la pura curiosidad) por las cuales podríamos desear conocer cuáles son estos dígitos restantes. Lo primero que hemos de hacer, por lo tanto, es ampliar las capacidades aritméticas del LOGO de modo que pueda calcular con una precisión mayor de siete cifras.

Para simplificar el asunto, sólo vamos a considerar los enteros positivos. Representaremos a los enteros como listas (de modo que representaremos 1.234.567 como [1 2 3 4 5 6 7]). Los dos procedimientos siguientes realizarán la suma de este tipo de números. Probemos con PRINT SUMALARGA[1 2 3] [5 6 9]; el resultado será [6 9 2]:

```
TO SUMALARGA :X :Y
  OUTPUT SUMALARGA1 :X :Y 0
END

TO SUMALARGA1 :X :Y :LLEVO
  IF(ALLOF (EMPTY?:X)(EMPTY?:Y)(:LLEVO
  =0))THEN OUTPUT[]
  TEST EMPTY? :Y
  IFTRUE IF :LLEVO=0 THEN OUTPUT :X ELSE
  OUTPUT SUMALARGA1 :X [1] 0
  TEST EMPTY? :X
  IFTRUE IF :LLEVO=0 THEN OUTPUT :Y ELSE
  OUTPUT SUMALARGA1 [1]:Y 0
  MAKE "SUMA(LAST :X)+(LAST :Y)+:LLEVO
  OUTPUT LPUT REMAINDER :SUMA 10
  SUMALARGA1·BUTLAST :X
  BUTLAST :Y QUOTIENT :SUMA 10
END
```



Estos procedimientos funcionan de forma muy similar a la que utilizaríamos si hiciéramos las sumas en papel, sumando desde la izquierda e incorporando cualquier número arrastrado desde la columna anterior.

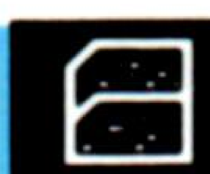
La resta es un proceso similar. No obstante, hemos incluido una rutina para suprimir de la respuesta los ceros no significativos, de modo que no acabemos con resultados tales como [0 0 0 7 8].

```
T RESTALARGA :X :Y
  OUTPUT QUITARCEROS RESTALARGA1 :X:Y 0
END

TO RESTALARGA1:X :Y :PIDO
  IF(ALLOF (EMPTY?:X)(EMPTY?:Y)(:PIDO=
  0))THEN OUTPUT [0]
  TEST EMPTY? :Y
  IFTRUE IF :PIDO=0 THEN OUTPUT :X
  ELSE OUTPUT RESTALARGA1 :X[1]0
  IF EMPTY? :X THEN PRINT [LO SIENTO,NO
  PUEDO MANIPULAR RESULTADOS
  NEGATIVOS] TOPLEVEL
  MAKE "DIFF(LAST :X)–(LAST :Y) – :PIDO
  IF:DIFF<0 THEN OUTPUT LPUT (10+:DIFF)
  RESTALARGA1 BUTLAST :X BUTLAST :Y 1
  OUTPUT LPUT :DIFF RESTALARGA1 BUTLAST
  :X BUTLAST :Y 0
END

TO QUITARCEROS :X
  IF EMPTY? :X THEN OUTPUT [0]
  IF NOT ((FIRST :X)=0)THEN OUTPUT :X
  OUTPUT QUITARCEROS BUTFIRST :X
END
```

La multiplicación larga es algo más complicada. La implementaremos utilizando la técnica que normalmente se enseña en las escuelas. Por ejemplo, vamos a suponer que queremos multiplicar 123 por 338. El problema se divide en tres partes: primero multiplicamos 123 por 8; luego multiplicamos 123 por 330; y, por último, sumamos los dos resultados entre sí. Este método se basa en el hecho de que la segunda etapa se puede dividir en dos subetapas: primero se multiplica 123 por 33, y luego se coloca un cero al final del resultado. Multiplicar un número por 33 evidentemente implica el uso de la recursión. El procedimiento MULTLARGA controla esta estrategia general:



0!	1
1!	1
2!	2
3!	6
4!	24
5!	120
6!	720
7!	5,040
8!	40,320
9!	362,880
10!	3,628,800
11!	39,916,800
12!	479,001,600
13!	6,227,020,800
14!	87,178,291,200
15!	1,307,674,368,000
16!	20.922.789.888.000
17!	355.687.428.096.000
18!	6.402.373.705.728.000
19!	121.645.100.408.832.000
20!	2.432.902.008.176.640.000

Ian McKinnell

Factor explosivo

Los valores factoriales aumentan con pasmosa rapidez, como podemos apreciar aquí. Debido a que los valores se vuelven tan grandes, la mayoría de los ordenadores y calculadoras representarán los factoriales de números mayores de 12 en notación exponencial. Por lo tanto, el factorial de 12 se dará como 4.79E8, o 4.79×10^8. Se incrementa la exactitud si se reflejan todos los dígitos significativos

```
TO MULTLARGA :X :Y
   IF EMPTY? BUTLAST :Y THEN OUTPUT
   MULTLARGA1 :X LAST :Y 0
   OUTPUT SUMALARGA(MULTLARGA1 :X
   (LAST :Y)0)
   (LPUT "0 MULTLARGA :X BUTLAST :Y)
END
```

Los detalles que supone la multiplicación de una línea por un único dígito los lleva a cabo MULTLARGA1:

```
TO MULTLARGA1 :X :NUM :LLEVO
   TEST EMPTY? :X
   IFTRUE IF :LLEVO=0 THEN OUTPUT [] ELSE
   OUTPUT (LIST :LLEVO)
   MAKE "PROD (LAST :X)*:NUM+ :LLEVO
   OUTPUT LPUT REMAINDER :PROD 10
   MULTLARGA1 BUTLAST :X :NUM QUOTIENT
   :PROD 10
END
```

Para el cálculo de factoriales no necesitamos procedimientos que efectúen la división, pero usted puede ampliar el sistema para que también cubra la división.

Ahora contamos con un conjunto de primitivas para llevar a cabo aritmética con cualquier grado de precisión. La única limitación respecto al tamaño de los números que se pueden manipular es el espacio total de memoria disponible para el programa.

Introducción de modificaciones

Ahora podemos modificar nuestro programa factorial original para utilizar la nueva forma de multiplicación larga.

```
TO FACT :X
   IF FIRST :X=0 THEN OUTPUT [1]
   OUTPUT MULTLARGA (FACT RESTALARGA
   :X [1]):X
END
```

Para probarlo digite FACT[1 3]; obtendrá [6 2 2 7 0 2 0 8 0 0]. Sin embargo, existen problemas. El proceso de cálculo es lento y (en el Commodore 64) el factorial más grande que obtuvimos antes de quedarnos sin memoria fue 34!, que tiene 39 dígitos (y calcularlo llevó algún tiempo).

La expresión de números grandes en forma de listas puede parecer inusual, pero podemos modificar el programa para que supere este problema traduciendo una y otra vez nuestra notación normal a la forma de lista. Empleamos dos procedimientos (EXPLOSION e IMPLOSION) para hacerlo.

EXPLOSION 123 produce [1 2 3] e IMPLOSION [1 2 3] produce 123.

```
TO EXPLOSION :X
   IF EMPTY? :X THEN OUTPUT []
   OUTPUT (SENTENCE FIRST :X EXPLOSION
   BUTFIRST :X)
END

TO IMPLOSION :X
   IF EMPTY? :X THEN OUTPUT"
   OUTPUT (WORD FIRST :X IMPLOSION
   BUTFIRST :X)
END
```

```
.
..
...
000,
000.
263,
130,
836,
933,
693,
530,
167,
218,
012,
160,
000,
000
.
2
631
30836
9336935
301672180
12160000000
```


```
.
..
...
000,
000.
263,
130,
836,
933,
693,
530,
167,
218,
012,
160,
000,
000
.
2
631
30836
9336935
301672180
12160000000
```

```
.
..
...
000,
000.
263,
130,
836,
933,
693,
530,
167,
218,
012,
160,
000,
000
.
2
631
30836
9336935
301672180
12160000000
```


Árbol de números
Los árboles factoriales se generan utilizando el dígito de más a la izquierda de un valor factorial como la copa del árbol. Los dígitos subsiguientes se extraen del valor real, de izquierda a derecha, en grupos cuyos tamaños van aumentando ligeramente. Los grupos se colocan debajo del dígito que constituye la piedra angular de una forma de árbol, y en posición simétrica respecto al mismo. Este diagrama representa el factorial de 32. El valor se puede leer más fácilmente cuando los números se disponen en grupos de tres, como podemos observar

Estos procedimientos se basan en el hecho de que en LOGO los números se tratan como palabras. Utilizándolas, podemos ahora definir un procedimiento, F:

```
TO F :X
  PRINT IMPLOSION FACT EXPLOSION :X
END
```

que calculará el factorial de 13 en respuesta a la entrada: F13.

El resultado de este cálculo (6227020800) es un poco difícil de leer como tal. Es más normal insertar puntos (6.227.020.800), que hacen que resulte más fácil de comprender. Los siguientes procedimientos dividen la palabra en grupos de tres dígitos e insertan puntos.

```
TO AÑADIRPUNTOS :X
  IF ((CONTADOR :X)<4) THEN OUTPUT:X
  OUTPUT(WORD AÑADIRPUNTOS
  CADATRES :X", ULTIMOSTRES :X)
END

TO CADATRES :X
  OUTPUT BUTLAST BUTLAST BUTLAST :X
END

TO ULTIMOSTRES :X
  OUTPUT (WORD (LAST BUTLAST BUTLAST :X)
  (LAST BUTLAST :X) (LAST :X))
END
```

Debemos, asimismo, modificar F para que incorpore estos procedimientos:

```
TO F:X
  PRINT AÑADIRPUNTOS IMPLOSION FACT
  EXPLOSION :X
END
```

La utilización de F para imprimir los primeros 20 factoriales nos da una idea de la rapidez con que crecen en tamaño los factoriales (los resultados se pueden apreciar en la tabla que ofrecemos en la página anterior).

Habiendo obtenido los factoriales de una gama de números, podemos comenzar a "jugar" con nuestros resultados. Un matemático norteamericano, por ejemplo, en una ocasión tuvo la brillante idea de imprimir números factoriales grandes en forma de árboles en las tarjetas de Navidad que les envió a sus amigos.

No son muchos los factoriales que poseen el número adecuado de dígitos para ser impresos como árboles, pero los siguientes procedimientos funcionarán si ello fuera posible:

```
TO ARBOL :L
  ARBOL1 1 :L
END

TO ARBOL1 :NUM :L
  IF EMPTY? :L THEN STOP
  REPEAT ROUND(20 – :NUM/2) [PRINT1
  ESPACIO] IMPRESIONLINEA :NUM :L
  ARBOL1 :NUM+2 PODAR :NUM:L
END

TO ESPACIO
  OUTPUT CHAR 32
END

TO IMPRESIONLINEA :NUM :L
  IF :NUM=0 THEN PRINT "STOP
  PRINT1 FIRST :L
  IMPRESIONLINEA :NUM – 1 BUTFIRST :L
END

TO PODAR :NUM :L
  IF :NUM=0 THEN OUTPUT :L
  OUTPUT PODAR :NUM – 1 BUTFIRST :L
END
```

Nuevamente debemos modificar nuestro procedimiento controlador:

```
TO F :X
  ARBOL IMPLOSION FACT EXPLOSION :X
END
```

El diagrama ilustra 32! escrito en forma de árbol. Si tiene intención de explorar más estos árboles factoriales, quizá le interesará saber que hay sólo tres números menores de 32 cuyos factoriales se pueden escribir como árboles. De los factoriales mayores adecuados el siguiente es 59!

Complementos al LOGO

En todas las versiones LCSI, utilice:

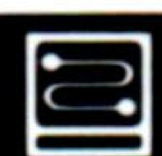
EMPTYP por EMPTY?
AND por ALLOF
TYPE por PRINT1

Existe, asimismo, una sintaxis diferente para IF. P. ej.:

```
IF :LLEVO=0[OUTPUT[]][OUTPUT (LIST :LLEVO)]
```

En lugar de QUOTIENT :X :Y emplee DIV :X :Y en el Spectrum y ROUND (:X/:Y) en el Atari.

En el Atari utilice SE por SENTENCE.



El toque FX del BBC Micro

Volvamos a analizar el sistema operativo del BBC Micro con mayor riqueza de detalles. Fijemos nuestra atención en las llamadas OSBYTE, que ofrecen una conveniente manera de acceder a muchas de las funciones del OS de esta máquina

En un primer momento, al estudiar cómo se accede al sistema operativo del BBC hablamos de un grupo de llamadas conocidas genéricamente como OSBYTE. Eran las que nos permitían modificar el comportamiento de varias partes del sistema operativo. Por ejemplo, la llamada OSBYTE *FX4,1 permite cambiar la manera en que el ordenador responde al pulsar una de las teclas del cursor en un teclado del BBC.

Si se piensa que el número de estas llamadas supera el centenar en la versión 1.2 del sistema operativo, no sorprende tanto el hecho de que ofrezcan una manera conveniente de acceder a muchas de las funciones del OS de dicho ordenador. Ya vimos que el empleo de las llamadas indirectas al sistema operativo nos protege contra posibles cambios en la configuración del software y del hardware de la máquina; la OSBYTE nos proporciona el método más importante de empleo de las rutinas del sistema operativo.

Antes de avanzar en el examen de OSBYTE, hay que tener en cuenta que con una máquina provista de una vieja versión 0.1 varias de las llamadas OSBYTE de las que aquí hacemos referencia no están soportadas.

Se comprueba el tipo de versión de una máquina BBC digitando sencillamente *HELP y RETURN.

Veamos primero cómo se emplea OSBYTE desde el BASIC y desde el lenguaje máquina. Como la mayoría de las llamadas del sistema operativo del BBC, también OSBYTE está vectorizada. Su vector se encuentra en las direcciones &20A y &20B.

Más abajo mostramos los modos de realizar una llamada OSBYTE. Todos ellos ejecutan la llamada OSBYTE antes mencionada, *FX4,1:

Desde BASIC usando *FX	Desde BASIC por medio de USR	En código máquina
*FX4,1	A%=4:X%=1:D%=USR(&FFF4)	LDA #4 LDX #1 JSR &FFF4

De estos tres ejemplos aparece claro que la dirección a la que hacemos las llamadas a las rutinas OSBYTE es la &FFF4, y los parámetros se pasan en la llamada OSBYTE en los registros A, X e Y del procesador 6502. Cuando se asigna un valor a A%, a X% y a Y% en BASIC y posteriormente se llama a una rutina del lenguaje máquina por medio de USR o CALL desde el BASIC, el programa en código máquina llamado será introducido con los registros A, X e Y del procesador conteniendo los valores de las variables A%, X% e Y%, respectivamente.

El empleo de USR en el segundo ejemplo nos permite obtener un resultado del programa en código máquina y devolverlo a una variable en BASIC. Lo cual es práctico con algunas llamadas OSBYTE, ya que éstas pueden devolver información al BASIC. De ello nos ocuparemos más adelante.

En el tercer ejemplo, empleamos un breve programa en código máquina para hacer una llamada OSBYTE; se trata de una sencilla carga de los registros necesarios con los valores adecuados y de la llamada OSBYTE a su dirección de llamada, la &FFF4. Estos ejemplos ilustran también la correspondencia existente entre los parámetros pasados al sistema operativo con la llamada FX (o sea, 4 y 1) y los parámetros pasados a los registros A, X e Y cuando llamamos las rutinas OSBYTE por medio del lenguaje máquina o la llamada USR.

Independientemente del modo empleado para llamar a la rutina OSBYTE, el contenido del registro A especificará siempre cuál de las muchas rutinas es la elegida. Los registros X e Y se utilizan entonces para pasar parámetros a la rutina mencionada del OSBYTE.

Algunas llamadas a OSBYTE no necesitan parámetros; otras sólo necesitan un parámetro que se pasará en el registro X; y las menos necesitan dos, pasados en los registros X e Y.

He aquí algunos ejemplos de estas llamadas a OSBYTE:

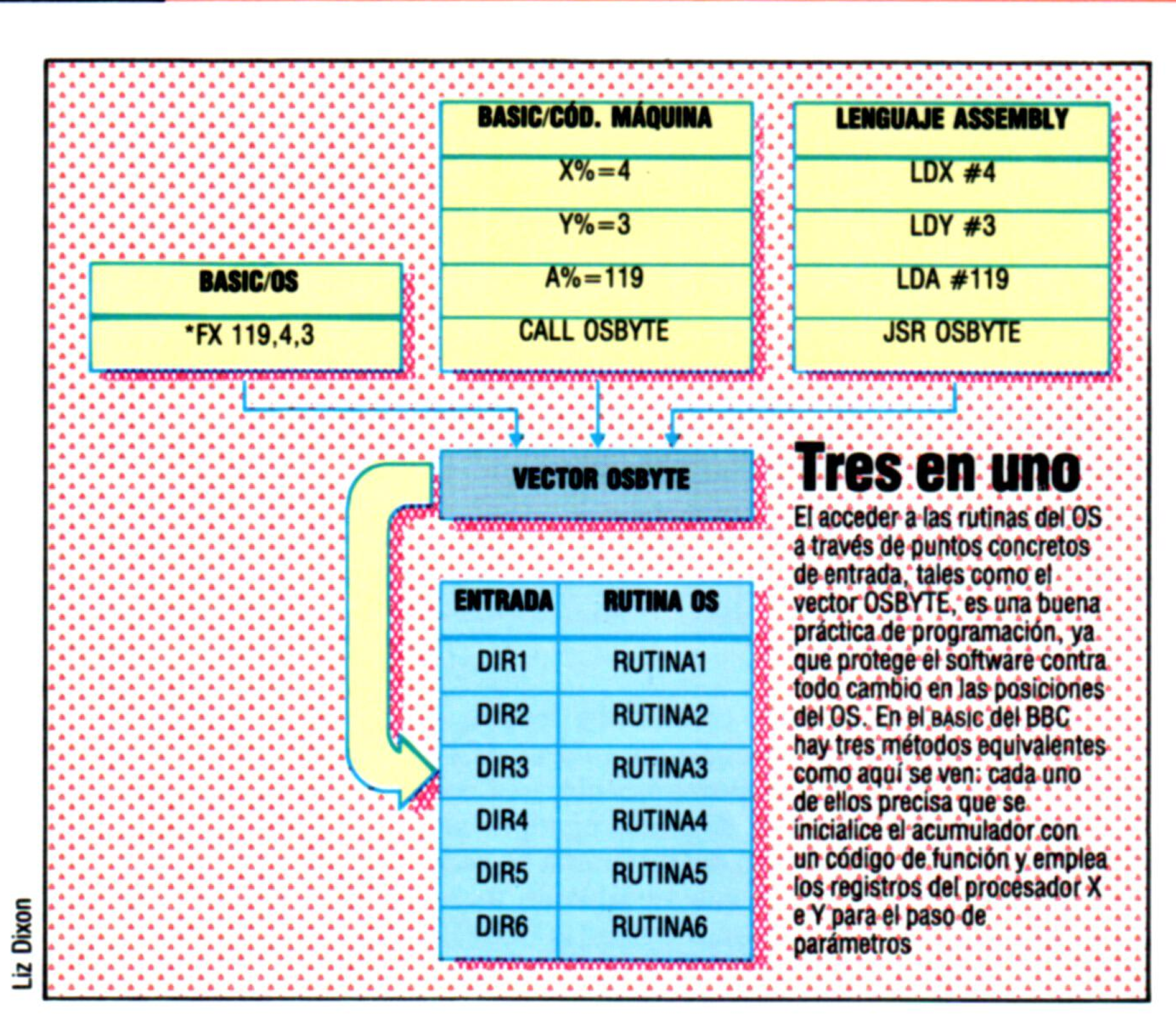
Sin parámetros	Un parámetro	Dos parámetros
*FX0	*FX4,1	*FX151,96,200

Hay que hacer algunas observaciones sobre cómo se llaman las rutinas OSBYTE por medio de *FX. La primera es que OSBYTE desconoce completamente las variables en BASIC, tales como las instrucciones *.

Si ejecutamos la codificación escrita más abajo obtendremos el siguiente mensaje de error Bad Command:

```
a=4:b=1
*FX a,b
```

Sin embargo, para obviar este problema se puede pasar la instrucción FX al OS por medio de OSCLI. Así:



Tres en uno
El acceder a las rutinas del OS a través de puntos concretos de entrada, tales como el vector OSBYTE, es una buena práctica de programación, ya que protege el software contra todo cambio en las posiciones del OS. En el BASIC del BBC hay tres métodos equivalentes como aquí se ven: cada uno de ellos precisa que se inicialice el acumulador con un código de función y emplea los registros del procesador X e Y para el paso de parámetros

```
10 DIM C 100
20 a=4
30 b=1
40 $C="*FX"+STR$a+","+STR$b
50 X%=C MOD 256
60 Y%=C DIV 256
70 CALL &FFF7
```

La segunda observación sobre las llamadas FX se refiere a que no se puede poner nada más en una línea de programa después de ellas.

Veamos:

```
*FX 4,1:PRINT "Oooops!"
```

Esta línea volverá a darnos el mensaje Bad Command: el sistema operativo considera los dos puntos y la instrucción PRINT como parte de la instrucción FX.

Ambos problemas se deben al hecho de que las llamadas *FX se pasan a través del intérprete de línea de instrucciones (CLI: *command line interpreter*) y no a través del intérprete BASIC, siendo el caso que CLI desconoce la manera de valorar las variables BASIC y de tratar líneas con varias instrucciones.

Como vimos, se pueden pasar parámetros al OSBYTE con los registros X e Y, y es posible también releer valores de algunas de las variables del sistema empleadas por el sistema operativo. Esto se hace con el empleo de la llamada USR o de una rutina en código máquina, como ya sabemos. Quizá usted encuentre el método del código algo más fácil de usar en el caso de que le interesen los resultados obtenidos del sistema operativo, ya que el valor obtenido con la llamada USR ha de ser decodificado para conseguir varios bits de información de ella.

Los parámetros son devueltos al BASIC con los registros X e Y, y también en algunas llamadas el flag de arrastre se utiliza para indicar una condición de error.

Obviamente los resultados que se devuelven de este modo dependerán de la llamada, es decir, del valor pasado al OSBYTE con el registro A. No todas las llamadas OSBYTE devuelven resultados al BASIC. De todas formas, muchas de ellas nos facilitan una útil información sobre el sistema operativo.

Información retornada

Son dos los tipos de información que puede retornar una llamada OSBYTE. El primero es el de los datos leídos de alguna parte del sistema, tal como la puerta para el usuario, el procesador de la voz, o las variables del sistema. Las llamadas OSBYTE que devuelven este tipo de información son conocidas como llamadas *de sólo lectura*. Un ejemplo típico de su empleo es la llamada OSBYTE con A=129. Esta llamada se utiliza en BASIC para implementar la función INKEY ().

Los registros X e Y deben establecerse para pasar el retardo temporal adecuado al sistema operativo. El registro X retiene el byte inferior de dicho retardo (en centisegundos) y el registro Y el byte superior. Así, para que esta llamada genere una espera de hasta un segundo hasta la pulsación de una tecla, el fragmento de código máquina que escribimos más abajo es válido. Los registros X e Y devuelven los valores; si el flag de arrastre está a cero, y el registro Y contiene asimismo un valor de 0, significa que la salida fue provocada por la pulsación de una tecla. El valor ASCII de esa tecla se encontrará en el registro X. Si Y contiene 255 y C está a 1, es señal de que no se pulsó tecla alguna en el período de tiempo especificado. Si C está a uno e Y contiene 27, indica que se pulsó la tecla Escape.

El siguiente fragmento en lenguaje máquina ilustra la manera de realizar esta llamada OSBYTE. Si el flag de arrastre está activado, a la vuelta de la rutina se hace una bifurcación a una rutina ulterior de tratamiento.

Esta rutina puede comprobar el valor contenido en el registro Y para saber si se pulsó una tecla o no, o bien se pulsó Escape.

```
1000  LDA    #129  /establece OSBYTE
1010  LDX    #100  /parámetros
1020  LDY    #0
1030  JSR    &FFF4 /realiza la llamada OSBYTE
1040  BCS    error
1050  RTS
1060  .error       /código para tratar un
                   error
```

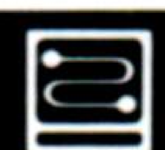
Otras llamadas OSBYTE, especialmente las que dan un valor a A entre 166 y 255, son llamadas de lectura y escritura a la vez, permitiéndonos leer o bien escribir ciertas variables del sistema en el sistema operativo.

Puede que usted empiece a preguntarse cómo sabe una llamada OSBYTE si le piden una operación de lectura o de escritura. Muy sencillo.

Para escribir un valor con una llamada OSBYTE, la llamada se realizará conteniendo el registro X el valor que deseamos que tal llamada escriba mientras el registro Y está puesto a 0.

Para leer un valor de una de estas variables del sistema, se pone X a 0 e Y contendrá 225. Después se hace la llamada.

Si va a haber una devolución, su valor se pondrá en los registros X e Y.



Empleo de las llamadas OSBYTE

Las llamadas OSBYTE son "cabos furrieles" del sistema operativo, que intervienen en un gran número de rutinas del sistema operativo. Los sistemas de ficheros, el teclado, Econet, las teclas Break y Escape, se ven afectados en mayor o menor medida por las llamadas OSBYTE.

El número de estas llamadas disponibles nos impide examinarlas una por una; aun así trataremos aquí las más útiles y menos estudiadas en el manual del usuario del BBC Micro.

Teclas de función: La orden *FX18 no tiene parámetros, pero es muy útil. Borra de la memoria las definiciones en curso de teclas de función, lo que resulta práctico a la hora de definir varias veces en el programa una tecla de función.

De *FX225 a *FX228: Si no se programó una tecla de función con una serie, se pueden utilizar estas llamadas para obligar a las teclas de función rojas a que nos den el valor ASCII. Por ejemplo, *FX225,n hará que la tecla de función f0 nos dé el código ASCII de n al pulsarla, y la f1 nos dará el de (n+1), etc. La *FX226 hace lo mismo cuando las teclas se pulsan juntamente con la techa Shift. La *FX227 sirve cuando se pulsan las teclas junto con la tecla Control, y la *FX228 lo hace cuando se pulsan a la vez Shift y Control además de una de función.

Funciones video y manejadores de VDU: Gran parte de la tarea de control de la visualización en el BBC Micro se realiza mediante la escritura de valores para los manejadores de VDU. Aun así, un par de llamadas OSBYTE vienen en nuestra ayuda en este contexto.

*FX19: Aunque no tenga parámetros, su utilidad se evidencia al programar gráficos en movimiento. Una vez ejecutada esta instrucción, el ordenador espera hasta que la siguiente "ventana" de visualización se haya escrito. Esto significa que el movimiento de los gráficos es menos "arbitrario". Ha de hacerse la llamada siempre que se necesite una pausa. Dado que la visualización se refresca 50 veces por segundo, es posible su empleo para generar retardos temporales o proporcionar interrupciones a la CPU.

*FX218: Es una llamada OSBYTE de lectura/escritura que nos informa de la longitud de la "cola" VDU (unidad de representación visual). Ya hemos explicado cómo algunos códigos VDU, cuando se envían a través de manejadores VDU, esperan que les sigan otros códigos. El número de bytes que aún esperan los manejadores de VDU en cualquier momento es el número de bytes de la cola VDU. De manera comprensiva, esta llamada se usa mejor para volver a leer información, y el resultado se retornará en el registro X.

*FX20: Esta llamada nos permite redefinir los caracteres en ASCII en el intervalo del 32 al 255, en vez del intervalo habitual y limitado de los caracteres definidos por el usuario. Así podemos redefinir, en Modos del 0 al 6, los caracteres a los que se accede desde el teclado normal. Para redefinir los caracteres en códigos ASCII dentro del intervalo del 32 al 128 debemos reservar memoria en nuestro espacio de trabajo del BASIC poniendo PAGE a un valor más alto que de costumbre. La redefinición de estos caracteres se hace con la llamada VDU 23. Los caracteres se redefinen en bloques de 32, y cada bloque exige 256 bytes de memoria. El único parámetro empleado se pasa al registro X. El manual del usuario proporciona más detalles sobre esta técnica.

Sonido: *FX210,n es una llamada útil para esos juegos generosos en ruidos "destruyeovnis". Permite diluir los efectos sonoros. El sonido se oye normalmente con *FX210,0 pero cualquier otro valor en el parámetro X matará este sonido.

*FX211 hasta *FX214: Son llamadas que controlan el sonido generado por CTRL-G o VDU7. Son de lectura/escritura. *FX211,n controla el canal del chip de sonido en el que se genera el sonido VDU7.*FX12,n controla el volumen del sonido generado o el envoltorio empleado para generar éste. El volumen se codifica de la misma manera que el parámetro de amplitud en la orden SOUND del BASIC (o sea, −15 es muy potente, 0 no se oye y números positivos son números de envoltorio). El valor pasado en el registro X en la llamada FX es dado por (n−1)*8, donde n es el parámetro. *FX213,n controla el tono *(pitch)* de la nota generada por el VDU7; *FX214,n controla la duración de la nota generada y tocada por VDU7.

Tecla Escape: *FX229,n permite a la tecla Escape "desconectarse". *FX229,1 desactiva la tecla Escape y *FX229,0 le devuelve su efecto habitual.

*FX220,n permite al usuario establecer una tecla para generar el evento de Escape, siendo n el código ASCII de la tecla que se desea actúe por Escape. Así, *FX220,65 produce el evento Escape cada vez que se pulsa la tecla A.

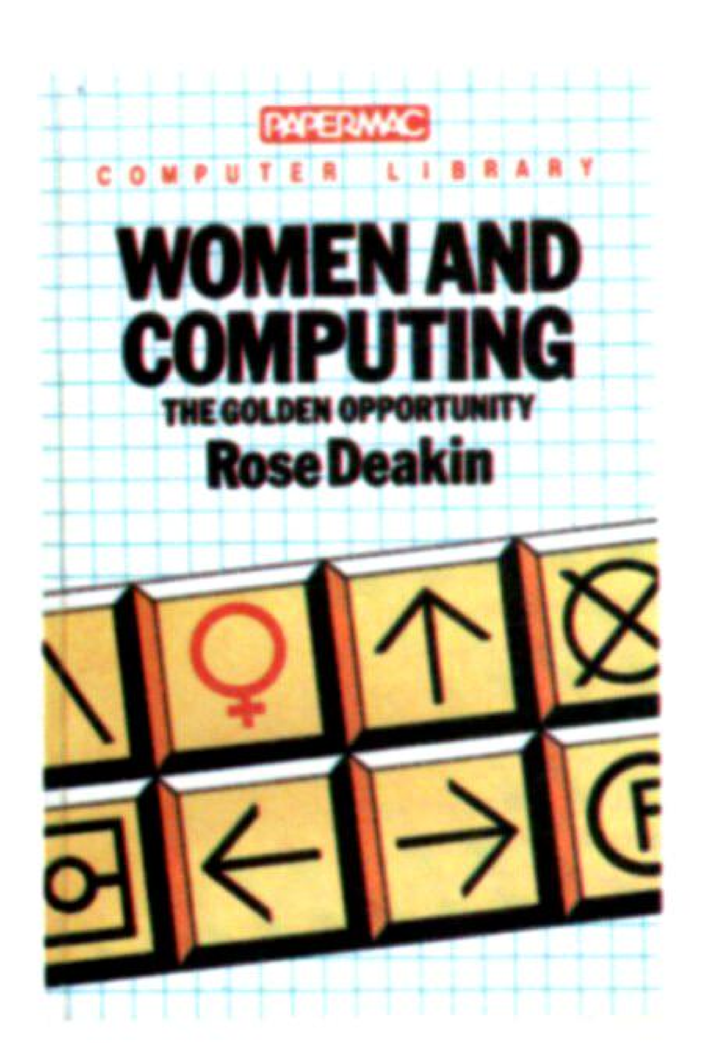
Buffers: *FX21,n disuelve, o vacía, un buffer (o memoria de paso). La operación consiste en descartar cualquier byte que se encuentre en el buffer. Por ejemplo, el vaciado del buffer del teclado desecha todas las teclas pulsadas que hayan podido acumularse mientras se ejecutaba un programa. Si usted prueba una orden GET cuando se tengan varias "teclas" sin procesar en el buffer del teclado, la orden GET tomará una tecla del buffer en vez de esperar a que se pulse una tecla. El vaciado de un buffer sonoro concluye la generación de sonido en ese determinado canal, aun habiendo otras instrucciones de sonido esperando a que se pulsen teclas. El buffer con el que se operará depende del parámetro X (véase cuadro al margen en esta página).

*FX138,n,m: Inserta el valor m en el buffer número n. Los números de los buffers son los mismos que para *FX21. Así, *FX138,0,65 introducirá la letra A en el buffer de teclado.

Todas estas llamadas OSBYTE se pueden realizar, como es obvio, pasando los parámetros que correspondan a los registros A, X e Y y haciendo la llamada a la subrutina conforme se explicó anteriormente.

Concluimos así el estudio de las principales funciones que quedan a cargo de las llamadas OSBYTE. Habría otras más por ser examinadas detenidamente (y se pueden encontrar en el manual), pero las técnicas son las mismas que las que acabamos de describir.

Parámetro X	Buffer operado por
0	Teclado
3	Impresora
4	Canal sonido 0
5	Canal sonido 1
6	Canal sonido 2
7	Canal sonido 3



"Women and computing" (Las mujeres y la informática), de Rose Deakin, Papermac, 1984

Los ordenadores transformarán la vida de las personas y de las naciones de la misma forma en que la han cambiado la rueda y el libro. Los dos libros que comentamos reflexionan, desde ópticas diferentes, acerca de esta evidencia cada vez más clara

"Women and computing"

"¿Por qué no animar a las mujeres? El director masculino es terrible; ha estado burlándose de la industria durante los últimos 20 años." Las palabras se le atribuyen a un profesor de psicología de la organización, pero tales ideas están muy ligadas al propio pensamiento de Rose Deakin, aunque ella es demasiado discreta para decirlo. Sin embargo, no pierde el tiempo deteniéndose en los hombres ni en la guerra de sexos; su preocupación es lograr que las mujeres vean la informática como una fuente de empleo y a los ordenadores como equipos industriales.

Su enfoque es tranquilo y desapasionado, presumiblemente fruto de su carrera como asistenta social, asesora de ventas de ordenadores y escritora. El libro se lee como un informe de viabilidad elaborado por un buen analista: la informática es una oportunidad que evidentemente las mujeres no están aprovechando. Deakin considera la educación, el empleo y la promoción como los remedios a largo plazo, y la ayuda propia y la empresa como los objetivos a corto plazo; en resumidas cuentas, las mujeres han de actuar ahora para sacar provecho del nuevo mercado.

"Es posible introducirse en la informática con pocas cualificaciones y poca experiencia, y sin capacidad ni inteligencia inauditas."

También hay, sin embargo, humor e ingenio. Deakin se complace en describir su sensación de regocijo tras dar un conferencia a 200 funcionarios públicos de gran antigüedad sobre administración de bases de datos, sólo unos pocos meses después de haber aprendido el significado del término, y está justificadamente orgullosa del libro al cual sirvió de base la conferencia.

Son de especial interés tres capítulos, en que se combinan sus virtudes personales y profesionales. Éstos describen las diferentes rutas que tomaron ella y otras siete mujeres en el campo de la informática, y demuestran de forma categórica su tesis: que las aptitudes de la mujer para organización y comunicación, y su capacidad para el pensamiento lúcido y el trabajo arduo la convierten en un usuario ideal de ordenadores.

"[Los investigadores señalan que]... las chicas hacen las sugerencias adecuadas para resolver el problema en cuestión; éstas son desoídas por los varones, que luego necesitan tres intentos para conseguirlo."

"Micro revolution revisited"

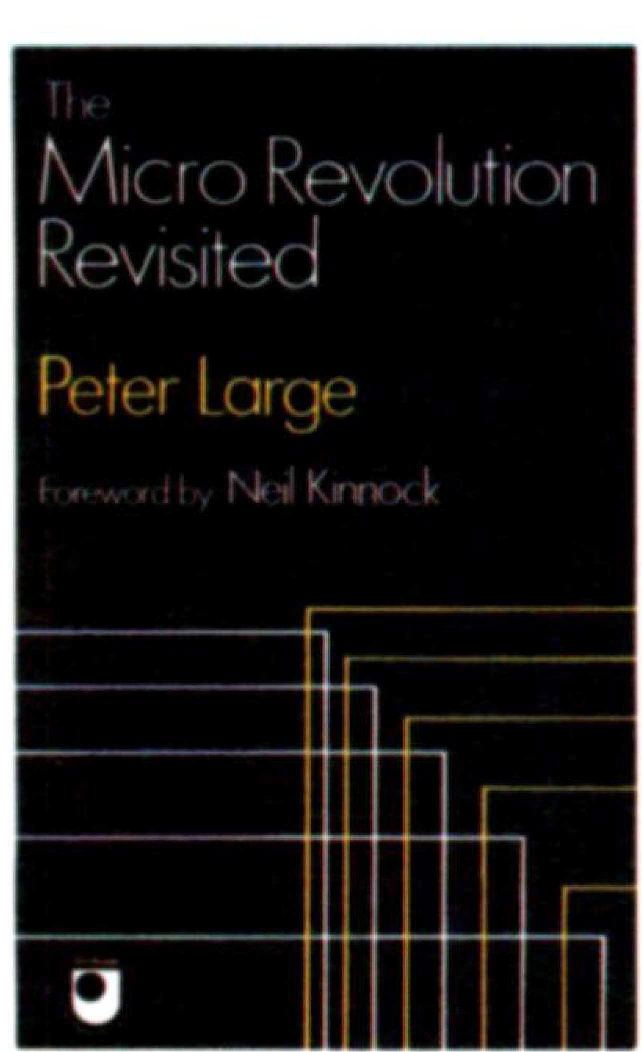
Un prólogo de Neil Kinnock, actual líder del Partido Laborista británico, el sello Set Book de la Open University de la cubierta y un corresponsal del diario *Guardian* como autor, son las credenciales de este libro. Éstas implican que la obra posee un interés social, es optimista, muy profesional y está bien documentada. El tema del libro es el desafío que representa para el *establishment* social el avance hacia la era de la informática.

"Al fin y al cabo, los ordenadores poseen una lógica matemática rígida: las personas, afortunadamente, no."

A partir de la confusa historia de los primeros 40 años de la informática, Large se detiene en los cinco peligros mortales de la informatización irreflexiva: delito, ineficacia, ignorancia, desempleo y totalitarismo. Bien podría haber añadido redundancia: este libro es, como mínimo, la sexta revisión de *The micro revolution*, escrito en 1980. Aunque comenzó proclamando con gran entusiasmo la nueva Revolución Industrial, ahora llama vigorosamente la atención sobre la posibilidad de volver a repetir los errores de la primera. Describe los artilugios y artefactos con entusiasmo y experiencia, desarrollando mientras tanto su *leitmotiv*: la vulnerabilidad de la sociedad industrial a los efectos de la informática sobre la producción, el empleo, la educación y la comunicación.

Large se muestra decididamente entusiasta respecto a las posibilidades de la tecnología, pero no sueña con que nuestra sociedad sea capaz de afrontar la descualificación, la pérdida de empleos y la automatización. Al comienzo del análisis del libro sobre trabajo, comunicaciones, robótica y desarrollos futuros, describe un día imaginario en la vida de Jane y Joe Babbage, alrededor de 2014 d.C. Jane edita un periódico financiero internacional desde una playa de Cornualles, mientras Joe, que es médico, realiza sus visitas rutinarias a través de la red pública de videotexto por ordenador. Los magros ingresos que obtienen con trabajos tan interesantes se comparan con el elevado salario que percibe Nat, un criado para tareas diversas de 73 años de edad que aún sabe hacer trabajos manuales. Hacia el final del libro resulta difícil saber si Large anhela o teme este futuro que imagina, y en qué cifras relativas cree él que seguiremos el cómodo camino de Jane y Joe hacia el empobrecimiento de la clase media y el sendero de Nat hacia la era de la abundancia de la clase trabajadora.

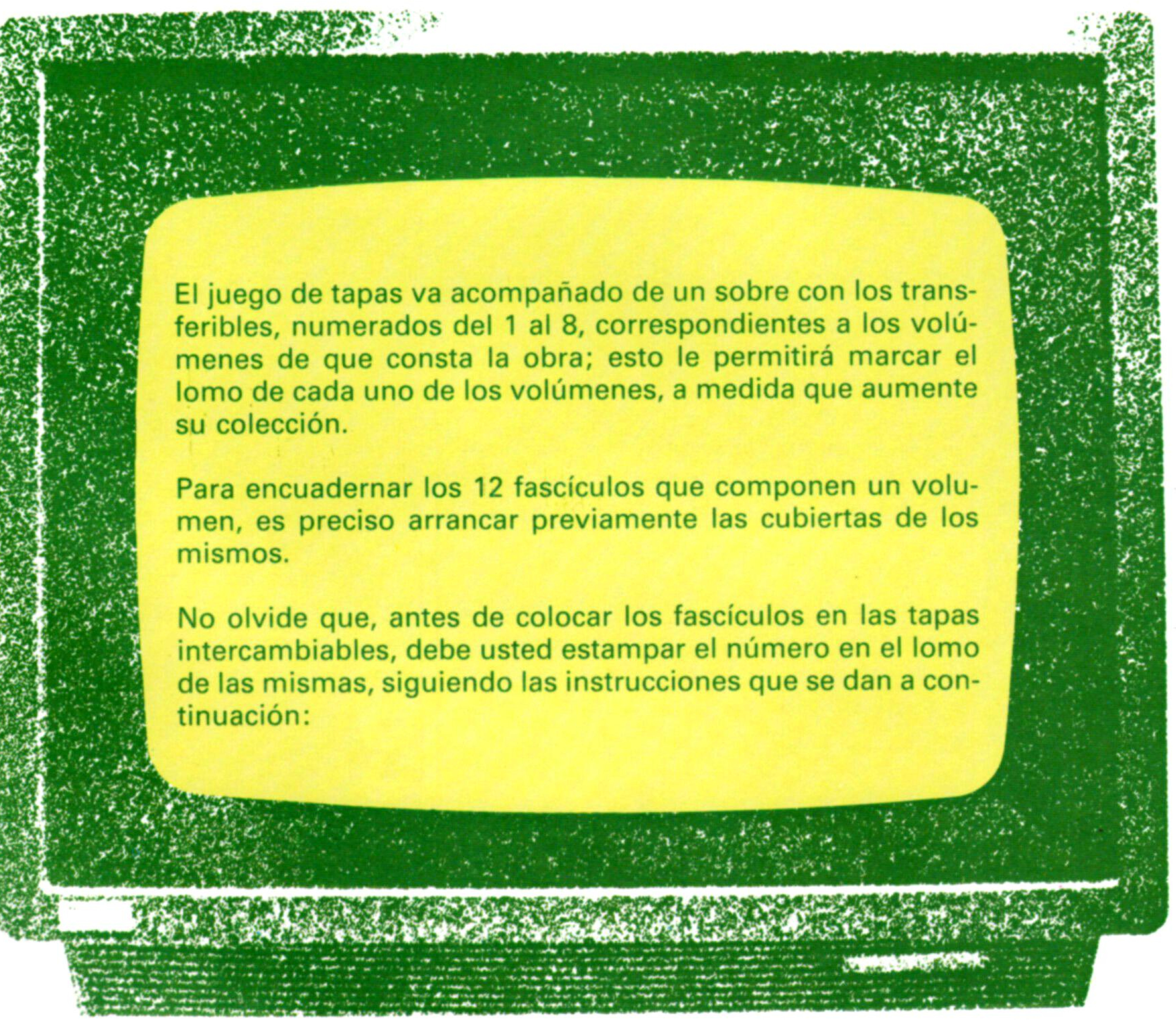
"Hemos dejado evolucionar nuestras máquinas en vez de volver a diseñarlas."



"The micro revolution revisited" (Retorno a la revolución del micro), de Peter Large, Frances Pinter, 1984



Con este fascículo se han puesto a la venta las tapas correspondientes al sexto volumen



El juego de tapas va acompañado de un sobre con los transferibles, numerados del 1 al 8, correspondientes a los volúmenes de que consta la obra; esto le permitirá marcar el lomo de cada uno de los volúmenes, a medida que aumente su colección.

Para encuadernar los 12 fascículos que componen un volumen, es preciso arrancar previamente las cubiertas de los mismos.

No olvide que, antes de colocar los fascículos en las tapas intercambiables, debe usted estampar el número en el lomo de las mismas, siguiendo las instrucciones que se dan a continuación:

1 **Desprenda la hojita de protección y aplique el transferible en el lomo de la cubierta, haciendo coincidir los ángulos de referencia con los del recuadro del lomo.**

2 **Con un bolígrafo o un objeto de punta roma, repase varias veces el número, presionando como si quisiera borrarlo por completo.**

3 **Retire con cuidado y comprobará que el número ya está impreso en la cubierta. Cúbralo con la hojita de protección y repita la operación anterior con un objeto liso y redondeado, a fin de asegurar una perfecta y total adherencia.**

Cada sobre de transferibles contiene una serie completa de números, del 1 al 8, para fijar a los lomos de los volúmenes. Ya que en cada volumen sólo aplicará el número correspondiente, puede utilizar los restantes para hacer una prueba preliminar.

PETICION DE FASCICULOS ATRASADOS

Si desea recibir algún fascículo atrasado o tapas, según las condiciones establecidas en el recuadro de la segunda página de cubierta («servicio de suscripciones y atrasados»), basta que rellene en LETRAS MAYUSCULAS este boletín y lo envíe a Editorial Delta, S.A., Paseo de Gracia, 88, 5.° - 08008 Barcelona.

NOMBRE
APELLIDOS
FECHA NACIMIENTO, DIA MES AÑO
PROFESION
DOMICILIO
N.° PISO ESCALERA
CODIGO POSTAL
POBLACION
PROVINCIA

OBRA:
N.os de fascículos atrasados que desea recibir:
....
....
N.° de tapas: ....
....



Ya están a su disposición, en todos los quioscos y librerías, las tapas intercambiables para encuadernar 12 fascículos de

mi COMPUTER

Cada juego de tapas va acompañado de una colección de transferibles, para que usted mismo pueda colocar en cada lomo el número de tomo que corresponda





10072
9 788485 822836